ROBUST AND EFFICIENT IMAGE-BASED 3D MODELING

BY

QINGXIONG YANG

DISSERTATION

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Electrical and Computer Engineering
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2010

Urbana, Illinois

Doctoral Committee:

Professor Narendra Ahuja, Chair
Professor Stephen Boppart
Professor David Forsyth
Assistant Professor Derek Hoiem
Professor Thomas Huang

ABSTRACT

In this dissertation, I report the progress towards building a robust and efficient 3D reconstruction system based on stereo vision. Stereo vision is known to be quite fragile in practice due to specular highlights, lack of texture, lighting variations, image blurring, etc. In this dissertation, I focus on exploiting the relationships between illuminants, surface reflection and shape to increase the robustness of stereo vision. I first present a new image transform for matching low-textured regions and then a robust solution for illumination chromaticity estimation based on a new correspondence matching invariant called Illumination Chromaticity Constancy. I next propose a new framework based on bilateral filtering and loopy belief propagation for simultaneous estimation of surface reflectance and shape with the assumption that the illumination chromaticity can be correctly estimated. Two new bilateral filtering algorithms with computational complexity invariant to filter kernel size and a new belief propagation with computational complexity invariant to the disparity search range are then presented to reduce the speed and memory cost.

ACKNOWLEDGMENTS

I would like to thank my adviser, Professor Narendra Ahuja, for his support and guidance. I would also like to thank my colleagues at the Computer Vision and Robotics Lab at University of Illinois at Urbana-Champaign, as well as the colleagues at HP Labs and Adobe Advanced Technology Labs, for their help and inspiration.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

3D reconstruction from stereo images is one of the most fundamental and extensively researched topics in computer vision. Stereo research has recently experienced somewhat of a new era, as a result of publicly available performance testing such as the Middlebury data set, which has allowed researchers to compare their algorithms against all the state-of-the-art algorithms. Nevertheless, stereo vision is known to be quite fragile in practice due to specular highlights, lack of texture, lighting variations, image blurring, etc. Additionally, while many stereo algorithms obtain high-quality results on Lambertian surfaces by performing global optimizations, today only simple correlation-based stereo algorithms are able to provide a dense (per pixel) depth map in real time.

In this dissertation, I focus on exploiting the relationships between illuminants, surface reflection and shape to increase the robustness of stereo vision. I also investigate the speed and memory issues of the existing stereo algorithms. New methods proposed have the potential to bring stereo vision applications to a new level.

I first propose a new image transform - epipolar distance transform, which captures the local image structure by computing the ratios of lengths along the epipolar lines. Theoretically, I extract the region boundaries such that the two endpoints of the line segment along the epipolar line can be located. I next compute the distance between the two endpoints and the distance between one endpoint and every pixel on the line segment. The ratio of the distances is invariant to affine transformation, and thus can be used as a matching invariant for stereo vision. Unlike image intensity/color, this invariant is robust for matching low-texture regions.

Based on a new correspondence matching invariant called Illumination Chromaticity Constancy, I next present a robust solution for illumination chromaticity estimation. Using a stereo image pair, the core of our method is the computation of a vote distribution for a number of illumination chromaticity hypotheses via correspondence matching. The hypothesis with the highest vote is accepted

as correct. Our experimental results show that our method is more robust than previous methods as more inputs are used.

I then propose a new framework for simultaneous estimation of surface reflectance and shape. A novel bilateral filtering based highlight removal method is presented to separate the diffuse and specular components using a single image. I then propose an iterative optimization approach for simultaneously estimating the depth and diffuse reflection via multi-view stereo matching and depth map fusion. I iteratively reduce reflection separation errors due to saturation, and depth estimation error due to incorrect reflectance estimates.

Two new bilateral filtering algorithms with computational complexity invariant to filter kernel size are then proposed. By showing that a bilateral filter can be decomposed into a number of constant time spatial filters or can be approximated using a linear combination of the original image, the Gaussian filtered responses of its powers, and their products, our methods yield a new class of constant time bilateral filters that can have arbitrary spatial and arbitrary range kernels. As an edge preserving operator, our constant time bilateral filter has been successfully applied to the proposed highlight removal method and high-quality stereo matching.

I finally consider the problem of stereo matching using loopy belief propagation. Unlike previous methods which focus on the original spatial resolution, I hierarchically reduce the disparity search range. By fixing the number of disparity levels on the original resolution, our method solves the message updating problem in a time linearly related to the number of pixels contained in the image and requires only constant memory space. Given the trend toward higher-resolution images and wide-baseline stereo vision, stereo matching using belief propagation with a large number of disparity levels with the efficiency of a few makes our method future-proof.

CHAPTER 2

STEREO MATCHING USING EPIPOLAR DISTANCE TRANSFORM

2.1 Introduction

Computational stereo for extraction of three-dimensional scene structure has traditionally been, and continues to be, an active area of intense research interest [1], [2]. In the past decade, much of the community's efforts were focused on the specific problem of *disparity optimization*, producing a number of excellent optimization methods that significantly advanced the state of the art. The key objective of these optimization methods is to reduce the matching ambiguities introduced by low-textured regions, and they can be generally classified into three categories: local methods, global methods, and hybrid methods.

The best local methods known today are based on either joint bilateral filtering [3] or image segmentation [4]. Yoon and Kweon [3] aggregate the matching cost with respect to both the color similarity and geometric proximity. Zitnick et. al. [4] aggregate the matching cost within each image segment and the obtained disparity maps from different cameras located at different positions are then fused to give coherent estimates.

The most popular global methods are based on belief propagation (BP) [5], [6] or graph cuts [7]. Both methods are formulated in an energy-minimization framework [8], where the objective is to find a disparity solution that minimizes a global energy function.

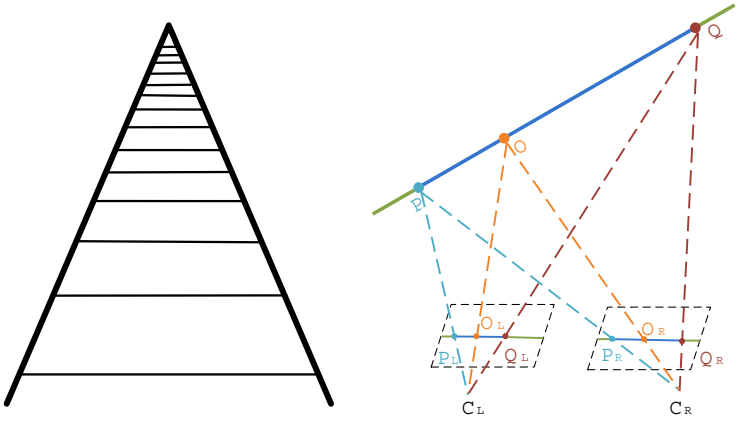
In low-textured regions, the lack of visual features makes matching a challenging problem. Local methods, which typically assume that the disparity values are the same for pixels inside the support window, do not work well on non-fronto-parallel surfaces which do not satisfy this assumption. Global methods are able to overcome this problem, but only when the size of the low-textured regions is relatively small. Several hybrid methods [9], [10], [11], [12] have been proposed to take advantage of both local and global optimization techniques. These meth-

ods assume planar surfaces and alternate between assigning pixels to 3D planes and refining the plane equations. A common problem with these techniques is that they rely on having accurate image segmentation, which may not always be robust. Other optimization methods, like Manhattan-world [13], [14], make more restricted assumptions or only extract vertical facades [15], [16].

All the above methods greatly advance the progress of stereo vision. However, they are struggling along on noisy matching cost values computed from the image intensities and ignoring the image structure from which robust matching invariants may be obtained.

This chapter proposes an image transform - epipolar distance transform, which captures the local image structure by computing the ratios of lengths along the epipolar lines. Theoretically, I extract the region boundaries such that the two endpoints of the line segment along the epipolar line can be located. I next compute the distance between the two endpoints and the distance between one endpoint and every pixel on the line segment. The ratio of the distances is invariant to affine transformation, and thus can be used as a matching invariant for stereo vision. Unlike image intensity/color, this invariant is robust for matching low-texture regions.

In practice, it is very difficult to obtain consistent region boundaries by applying segmentation to a stereo image pair. Instead, I compute the distance between a pixel and an endpoint (say the endpoint on the left) of the line segment containing the pixel by aggregating the contribution of the other pixels on the left along the epipolar line based on intensity similarity. Lower similarity corresponds to lower contribution. Specifically, if the epipolar line contains only black and white pixels, the contribution of every pixel can then be computed using a binary function. The summation of these binary values will then equal the distance to be computed. For real images, I compute each pixel's contribution using a Gaussian function, and the distances can be computed as an adaptive-weighted aggregation. This aggregation can be computed very fast using integral histograms [17]. Our implementation on 2.5 GHz Intel Core 2 Duo processor can achieve real-time speed for VGA-sized image, and over 500 frames per second on an NVIDIA 8800 GTX GPU. Besides the speed advantage and the robustness for matching low-textured regions, our method is also robust to the optical vignetting problem exhibited in practical stereo systems and keypoint detection and description for low-textured surfaces. Results on a variety of indoor and outdoor images demonstrate our claims.



(a) Railway. (b) Affine invariant.

Figure 2.1: An affine invariant.

2.2 Epipolar Distance Transform

The lack of texture and lens vignetting leads to ambiguities/errors in matching when the image intensity/color is used as the matching invariant. On the other hand, the geometric properties of the image segments, including the region area, boundary shape, and relative distances between regions, are more robust to the intensity variances. Ahuja [18] proposed an image transform to extract the image local structure by computing an attraction-force field over the image. This transform can be used to extract a multi-scale segmentation tree, which has been proven to be very efficient for object categorization [19]. However, this image transform is not affine invariant and cannot be used for stereo matching. Actually, very few quantitative properties of an object remain fixed when it is drawn in perspective. For instance, Figure 2.1 (a) is a perspective drawing of a railway track. The lengths and areas are not preserved. Even the ratios of lengths along a line are not preserved because the sleepers are equally spaced while they get closer and closer in the perspective drawing.

In this section, I present a new image transform - epipolar distance transform - to capture the image structure. This transform is invariant to affine transform and can be used as a matching invariant for stereo vision. Specifically, I use the ratios of lengths along the epipolar line. See Figure 2.1 (b). To make the problem simpler, I assume that the camera motion is pure translation, or equally the camera images are rectified such that the epipolar lines are scanlines. Let $\mathbf{P}_L\mathbf{Q}_L$ and $\mathbf{P}_R\mathbf{Q}_R$ be two corresponding line segments along the epipolar line on

camera C_L and C_R, $\mathbf{PQ}$ be the corresponding line segment in Euclidean 3-space $\mathbb{R}^3$, and $\mathbf{O}_L$ be any point inside $\mathbf{P}_L\mathbf{Q}_L$ with $\mathbf{O}_R$ its correspondence in camera C_R. I then obtain an affine invariant:

$$\frac{\| \mathbf{P}_R - \mathbf{O}_R \|}{\| \mathbf{P}_R - \mathbf{Q}_R \|} = \frac{\| \mathbf{P}_L - \mathbf{O}_L \|}{\| \mathbf{P}_L - \mathbf{Q}_L \|}. \tag{2.1}$$

Proof: Because $\mathbf{PQ}$ is a line in $\mathbb{R}^3$, $\mathbf{O}$ is a point inside $\mathbf{PQ}$, and the projection of $\mathbf{PQ}$ is an epipolar line on camera C_L, then the disparity values of their projections on camera C_L is a linear function of the x-axis values (denote as x) of their image projections:

$$D(x) = kx + b, \tag{2.2}$$

where k and b are two constants. Letting $x_R^{\mathbf{P}}$,$x_R^{\mathbf{O}}$, $x_R^{\mathbf{Q}}$, $x_L^{\mathbf{P}}$,$x_L^{\mathbf{O}}$, $x_L^{\mathbf{Q}}$ be the x-axis values of $\mathbf{P}_R$, $\mathbf{O}_R$, $\mathbf{Q}_R$, $\mathbf{P}_L$, $\mathbf{O}_L$, $\mathbf{Q}_L$,

$$x_R^{\mathbf{P}} = x_L^{\mathbf{P}} - D(x_L^{\mathbf{P}}), \tag{2.3}$$
$$x_R^{\mathbf{O}} = x_L^{\mathbf{O}} - D(x_L^{\mathbf{O}}), \tag{2.4}$$
$$x_R^{\mathbf{Q}} = x_L^{\mathbf{Q}} - D(x_L^{\mathbf{Q}}), \tag{2.5}$$

I then obtained

$$\frac{\| \mathbf{P}_R - \mathbf{O}_R \|}{\| \mathbf{P}_R - \mathbf{Q}_R \|} = \frac{|x_R^{\mathbf{P}} - x_R^{\mathbf{O}}|}{|x_R^{\mathbf{P}} - x_R^{\mathbf{Q}}|}, \tag{2.6}$$
$$= \frac{|x_L^{\mathbf{P}} - D(x_L^{\mathbf{P}}) - (x_L^{\mathbf{O}} - D(x_L^{\mathbf{O}}))}{|x_L^{\mathbf{P}} - D(x_L^{\mathbf{P}}) - (x_L^{\mathbf{Q}} - D(x_L^{\mathbf{Q}}))|}, \tag{2.7}$$
$$= \frac{|x_L^{\mathbf{P}} - (k(x_L^{\mathbf{P}}) + b) - (x_L^{\mathbf{O}} - k(x_L^{\mathbf{O}}) - b)|}{|x_L^{\mathbf{P}} - (k(x_L^{\mathbf{P}}) + b) - (x_L^{\mathbf{Q}} - k(x_L^{\mathbf{Q}}) - b)|}, \tag{2.8}$$
$$= \frac{|(1-k)x_L^{\mathbf{P}} - (1-k)x_L^{\mathbf{O}}|}{|(1-k)x_L^{\mathbf{P}} - (1-k)x_L^{\mathbf{Q}}|}, \tag{2.9}$$
$$= \frac{|x_L^{\mathbf{P}} - x_L^{\mathbf{O}}|}{|x_L^{\mathbf{P}} - x_L^{\mathbf{Q}}|}, \tag{2.10}$$
$$= \frac{\| \mathbf{P}_L - \mathbf{O}_L \|}{\| \mathbf{P}_L - \mathbf{Q}_L \|}. \tag{2.11}$$

Actually, the ratio of lengths along any line in C_L is preserved under affine transformations. However, it is hard to find its correspondences in C_R except along the scanlines (epipolar lines) in practice. For instance, a vertical line in C_L may not be also a vertical line in C_R. The epipolar geometry, however, guarantees

that the correspondence of a line segment along a scanline in C_L also lies on the same scanline in C_R.

To compute the matching invariant for each image point $\mathbf{O}_L$ (or $\mathbf{O}_R$), I have to detect two endpoints of the line segment: $\mathbf{P}_L$ and $\mathbf{Q}_L$ (or $\mathbf{P}_R$ and $\mathbf{Q}_R$). Alternatively, I can segment the image into regions, then compute the matching invariant for each image point $\mathbf{O}_L$ as an integration of the pixels on the line segment along the scanline. Let the scanline width be $\mathbf{w}$,

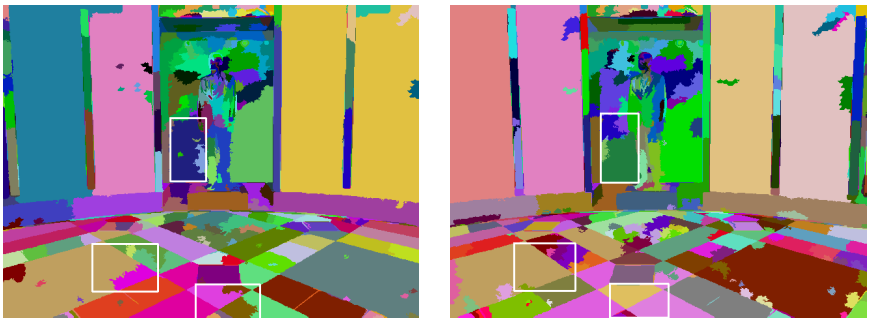
$$\| \mathbf{P}_L - \mathbf{O}_L \| = \int_0^{x_L^{\mathbf{O}}} sign(x, x_L^{\mathbf{O}}) dx, \tag{2.12}$$

$$\| \mathbf{P}_L - \mathbf{Q}_L \| = \int_0^{\mathbf{w}-1} sign(x, x_L^{\mathbf{O}}) dx, \tag{2.13}$$

where x is the x-axis value of a pixel $\mathbf{x}$ along the scanline, and

$$sign(x, x_L^{\mathbf{O}}) = \begin{cases} 1 & \mathbf{x} \text{ and } \mathbf{O}_L \text{ are in the same segment,} \\ 0 & \text{otherwise.} \end{cases}$$

However, image segmentation is known to be non-robust. For instance, the de-



(a) Left. (b) Right.

Figure 2.2: Meanshift segmentation. The white boxes indicate where the system obtains inconsistent segmentation results on the left and right images.

tected line segment $\mathbf{P}_L\mathbf{Q}_L$ is likely to be separated into multiple line segments in the other camera. Such examples are provided in Figure 2.2. The EDISON system (Meanshift segmentation) [20] with default parameters is used to segment the left and right camera images in Figure 2.6 (a) and (b) on p. 14, and the results are presented in Figure 2.2. The white boxes indicate where the system obtains inconsistent segmentation results on the left and right images. Also, color segmentation is generally slow. I thus adopt a "soft segmentation" approach. Specifically, I re-

place the $sign()$ function in Eqn. (2.12) and (2.13) with a Gaussian function with standard deviation σ_R:

$$g(I(x), I(x_L^{\mathbf{O}})) = exp(-\frac{(I(x) - I(x_L^{\mathbf{O}}))^2}{2\sigma_R^2}), \tag{2.14}$$

where $I(x)$ and $I(x_L^{\mathbf{O}})$ are the intensity values of image point $\mathbf{x}$ and $\mathbf{O}_L$. The function $g(I(x), I(x_L^{\mathbf{O}}))$ increases with the intensity similarity between $\mathbf{x}$ and $\mathbf{O}_L$. Also note that σ_R should be as small as possible to suppress the contribution of pixels from different regions. If I set σ_R to 0, then it is the sign() function. However, to account for sensor noise, I set σ_R to 7 for all the 8-bit images used in our experiments. The Gaussian function is the approximation of the sign() function, but more robust in practice. As indicated in Eqn. (2.14), to make sure that the Gaussian integration with $\sigma_R > 0$ is also affine invariant, I have to assume that the correspondences of the image gradients computed from the left and right camera image are the same. The existence of noise in real images violates this assumption, but our experiments show that this technique is robust to noise when σ_R is sufficiently large, e.g., $\sigma_R = 7$ in our experiments.

Let the matching invariant be $\mathcal{F}$; for each image point $\mathbf{O}_L$, the matching invariant is actually an integration of the pixels along the epipolar line such that the pixels belonging to the same region are mutually attracted:

$$\begin{aligned}
\mathcal{F}(\mathbf{O}_L) &= \frac{\parallel \mathbf{P}_L - \mathbf{O}_L \parallel}{\parallel \mathbf{P}_L - \mathbf{Q}_L \parallel} = \frac{\int_{\max(0, x_L^{\mathbf{O}} - \sigma_S \mathbf{w})}^{x_L^{\mathbf{O}}} g(I(x) - I(x_L^{\mathbf{O}}))dx}{\int_{\max(0, x_L^{\mathbf{O}} - \sigma_S \mathbf{w})}^{\min(\mathbf{w}-1, x_L^{\mathbf{O}} + \sigma_S \mathbf{w})} g(I(x) - I(x_L^{\mathbf{O}}))dx} \\
&= \frac{\int_{\max(0, x_L^{\mathbf{O}} - \sigma_S \mathbf{w})}^{x_L^{\mathbf{O}}} exp(-\frac{(I(x) - I(x_L^{\mathbf{O}}))^2}{2\sigma_R^2})dx}{\int_{\max(0, x_L^{\mathbf{O}} - \sigma_S \mathbf{w})}^{\min(\mathbf{w}-1, x_L^{\mathbf{O}} + \sigma_S \mathbf{w})} exp(-\frac{(I(x) - I(x_L^{\mathbf{O}}))^2}{2\sigma_R^2})dx},
\end{aligned} \tag{2.15}$$

where σ_S is a scalar controlling the size of the local regions to be taken into account. It is hard to determine the value of σ_S at each pixel location. Larger σ_S is not robust to occlusion while smaller σ_S is invalid for large low-textured regions. Ideally, I should have large σ_S for low-textured regions and small σ_S for high-textured regions. In this chapter, I set σ_S to a constant value (0.03), and use Eqn. (2.15) to compute the matching invariant $\mathcal{F}$ at each pixel location, which is actually an image transform incorporating the duality of interior and edge-based descriptions of regions along the epipolar line. Nevertheless, our experiments on real images demonstrate that stereo matching using the transformed images with

constant σ_S can perform much better than standard intensity-based stereo matching methods for low-textured regions. See Section 2.4 for details.

2.3 Fast Image Transform Using Integral Histogram

In this section, I present a method enabling the image transform presented in Section 2.2 to be computed in real time for 8-bit images. This method is based on histogram. Assuming $\sigma_S = 1$, I first compute the histogram for each scanline and let it be $\vec{H}$. The denominator in Eqn. (2.15) can then be computed using $\vec{H}$:

$$\begin{aligned}\int_0^{\mathbf{w}-1} exp(-\frac{(I(x)-I(x_L^{\mathbf{O}}))^2}{2\sigma_R^2})dx &= \sum_{s=[0,255]} \vec{H}(s)exp(-\frac{(s-I(x_L^{\mathbf{O}}))^2}{2\sigma_R^2}), \\ &= \sum_{s=[0,255]} \vec{H}(s)g(s, I(x_L^{\mathbf{O}})). \qquad (2.16)\end{aligned}$$

Note that the use of histogram enables the computation at each pixel location to be independent of the scanline width $\mathbf{w}$. Also the computation of Gaussian function $g(s, I(x_L^{\mathbf{O}}))$ is very efficient as it has up to 256 values (for 8-bit images) which can be pre-computed and stored as a lookup table.

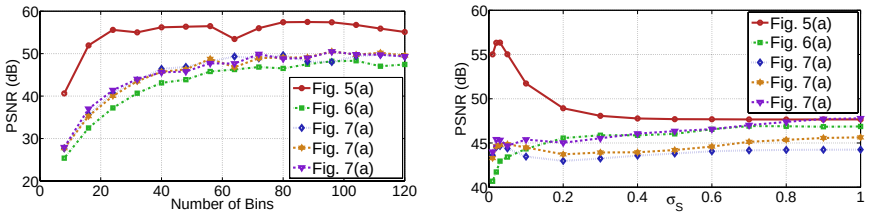
Traditional histogram is invalid for calculating the numerator in Eqn. (2.15) or the denominator when $\sigma_S < 1$, because only part of the pixels on the scanline are taken into account. Fortunately, the integral histogram [17] can be used. As the name suggests, the value at any point $\mathbf{O}_L$ in the integral histogram is just the sum of all the pixels to the left of $\mathbf{O}_L$. Let the integral histogram computed from the scanline be $\vec{H}^I$ which is a 2D array; then $\vec{H}^I(x_L^{\mathbf{O}}) - \vec{H}^I(x)$ is the histogram of the pixels with x-axis value ranges from x to $x_L^{\mathbf{O}}$. Using integral histogram, the computation of the image transform presented in Eqn. (2.15) is independent of the scanline width $\mathbf{w}$:

$$\mathcal{F}(\mathbf{O}_L) = \frac{\sum_{s=[0,255]}(\vec{H}^I(x_L^{\mathbf{O}}, s) - \vec{H}^I(xmin, s))g(s, I(x_L^{\mathbf{O}}))}{\sum_{s=[0,255]}(\vec{H}^I(xmax, s) - \vec{H}^I(xmin, s))g(s, I(x_L^{\mathbf{O}}))}, \qquad (2.17)$$

$$xmin = \max(0, x_L^{\mathbf{O}} - \sigma_S\mathbf{w} - 1), \qquad (2.18)$$

$$xmax = \min(\mathbf{w} - 1, x_L^{\mathbf{O}} + \sigma_S\mathbf{w}). \qquad (2.19)$$

In practice, the number of bins used to construct the integral histogram $\vec{H}^I$ can be smaller than 256 (number of intensity levels) without significantly impacting the



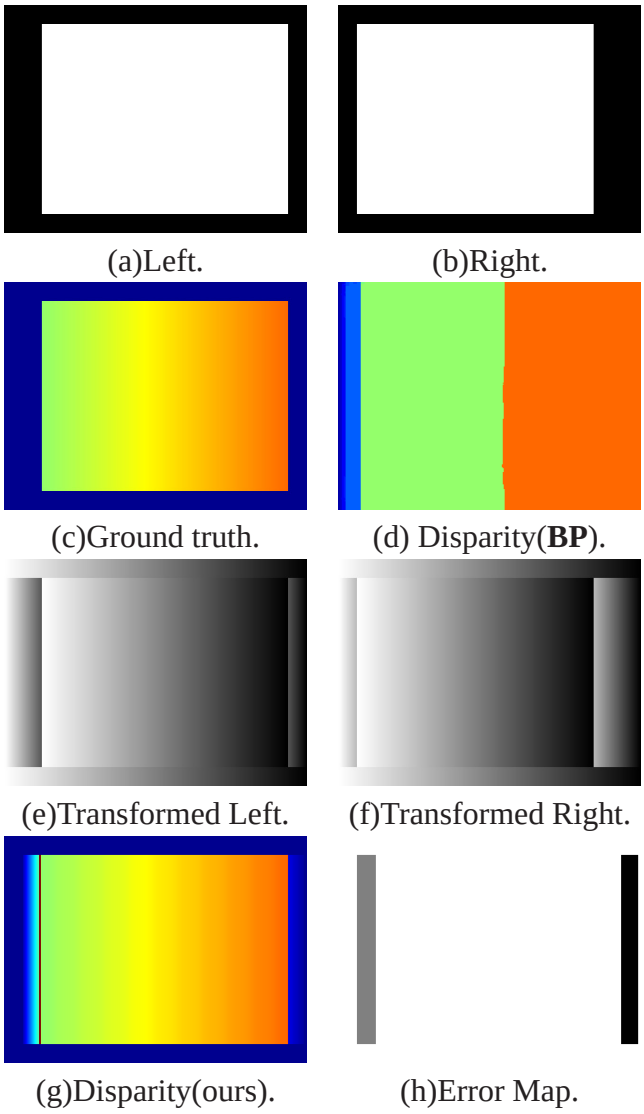
(a) (b)

Figure 2.3: Numerical evaluation of our fast image transform method using PSNR.

accuracy. The selection of the number of bins actually depends on the value of σ_R. Larger σ_R requires lower bins. In this chapter, I set $\sigma_R = 7$, and our experimental results show that using 32-bin histogram can achieve high peak signal-to-noise ratio (PSNR) as shown in Figure 2.3. The five curves in Figure 2.3 correspond to the five real images used in our experiments (Section 2.4). Figure 2.3 (a) shows that the performance of our method increases as the number of bins used in constructing integral histogram increases. Figure 2.3 (b) numerically evaluates the performance of our method with respect to σ_S when σ_R is set to 7, and the number of bins is set to 32. Note that the PSNR values are all over 40 dB. It is assumed [21] the PSNR values above 40 dB often correspond to almost invisible differences; thus, I used 32 as the number of bins in all our experiments.

2.4 Experiments

In this section, I present experiments on both synthetic and real images to demonstrate the effectiveness and robustness of our method. Section 2.4.1 experimentally proves that the proposed image transform is affine invariant using a synthetic data set. Section 2.4.2 numerically evaluates the performance of our image transform for stereo matching using two real indoor data sets, and Section 2.4.3 presents visual evaluation on three outdoor data sets. Finally, I show that our image transform is robust to keypoint detection and description for low-texture scenes in Section 2.4.4. All the real images used in our experiments are captured by a commercial stereo vision system: Point Gray Bumblebee XB3 stereo vision system [22]. Similar to the other systems, the lenses of Bumblebee XB3 stereo vision system exhibit optical vignetting to some degree, which causes problems



(a)Left. (b)Right.

(c)Ground truth. (d) Disparity(**BP**).

(e)Transformed Left. (f)Transformed Right.

(g)Disparity(ours). (h)Error Map.

Figure 2.4: Synthetic data set.

for stereo matching, especially for low-texture regions. All the experiments conducted on the real images use the same parameters. Specifically, I set σ_R to 7, σ_S to 0.03, and the number of bins to 32.

2.4.1 Stereo Matching Using a Synthetic Scene

This section presents our experimental results on a synthetic scene, which experimentally prove that the proposed epipolar distance transform is robust for matching low-textured regions. Figure 2.4 (a) and (b) are the synthetic left and right

images with ground-truth disparity map presented in (c). The disparity map obtained from intensity-based belief propagation [6] is presented in Figure 2.4(d). It is apparent that the intensity-based method is invalid for this scene due to the lack of texture. The transform presented in the chapter, however, captures the local region structure, and brings in variances to the low-texture regions. The transformed images are presented in Figure 2.4 (e) and (f). The disparity map computed from the transformed images is presented in Figure 2.4 (g). Since there is no noise, I do not use any local or global disparity optimization method. However, I use the quadratic polynomial interpolation method presented in [23] to obtain sub-pixel accuracy. The error map computed from the ground truth in Figure 2.4 (c) and (g) is presented in (h). The black and gray pixels have disparity error larger than 0.5, and the gray pixels are half-occluded, a situation that is not considered in this chapter. As can be seen, most of the pixels in Figure 2.4 (g) have disparity error less than half a pixel. In this experiment, I set σ_S to 1 because the area of the untextured region is almost the same as the whole image.

2.4.2 Numerical Evaluation Using Real Indoor Scenes

In this section, I numerically evaluate the performance of our image transform for stereo matching using two real indoor data sets with regions that are weakly textured. I first evaluate our method with a low-textured wall as shown in Figure 2.5 (a)-(b). I adjusted the camera to make sure that the wall is fronto-parallel such that the ground-truth z-depth values are the same for every pixel and can be manually measured. The ground-truth disparity map is presented in Figure 2.5 (c). Figure 2.5 (d) and (e) are the transformed images of (a) and (b), respectively. Figure 2.5 (f) and (i) are the disparity maps obtained by applying standard belief propagation to the camera images and the transformed images, and the corresponding disparity error maps are presented in (g) and (h), respectively. As can be seen in Figure 2.5 (g), most of the pixels are black, which corresponds to disparity error lager than 1. These errors are corrected using our method as shown in Figure 2.5 (h) except for the border occlusion in the left which is not considered in this chapter.

Figure 2.6 presents the reconstruction results for a cylinder-like lobby. The camera was placed in the center of the lobby and the angle of the camera was adjusted such that the z-depth values of the pixels in each column are the same and can be measured. I then manually segmented the reference image in Figure

2.6 (a) and applied plane fitting to each segment using the keypoints detected and matched using SIFT [24] except for the wall surfaces. The obtained disparity map is presented Figure 2.6 (c). The disparity error maps presented in Figure 2.6 (g) and (h) show that most of the errors (black pixels) caused by intensity-based stereo matching methods are corrected using our method. Note that the wall is not a planar surface, which violates the assumption made in Section 2.2. However, in practice, I set σ_S to 0.03, such that only the geometric properties of local patches are taken into account. Since the wall surface is locally planar, our image transform is still valid for stereo matching.

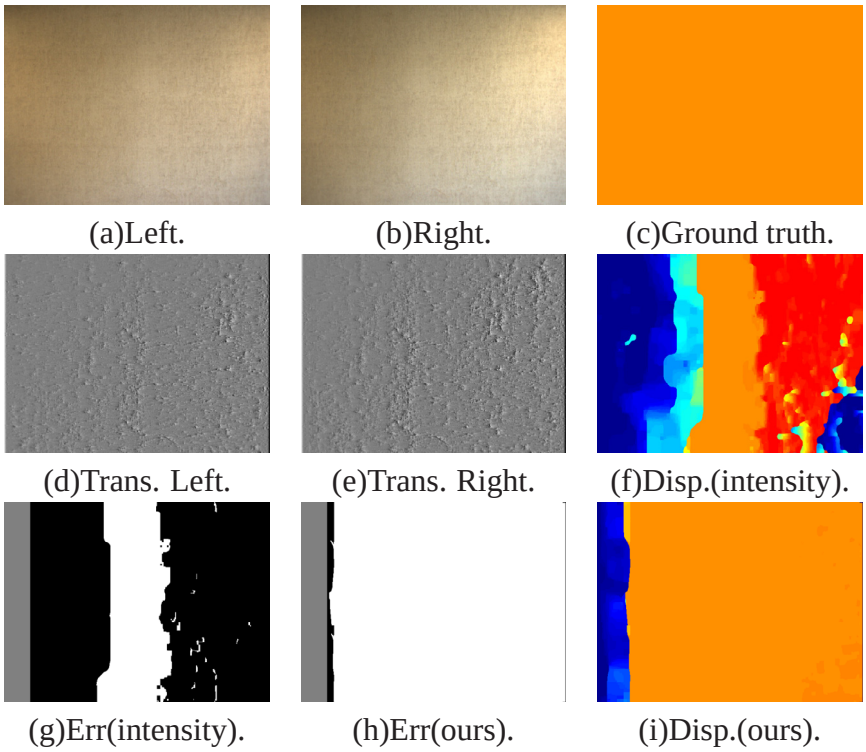
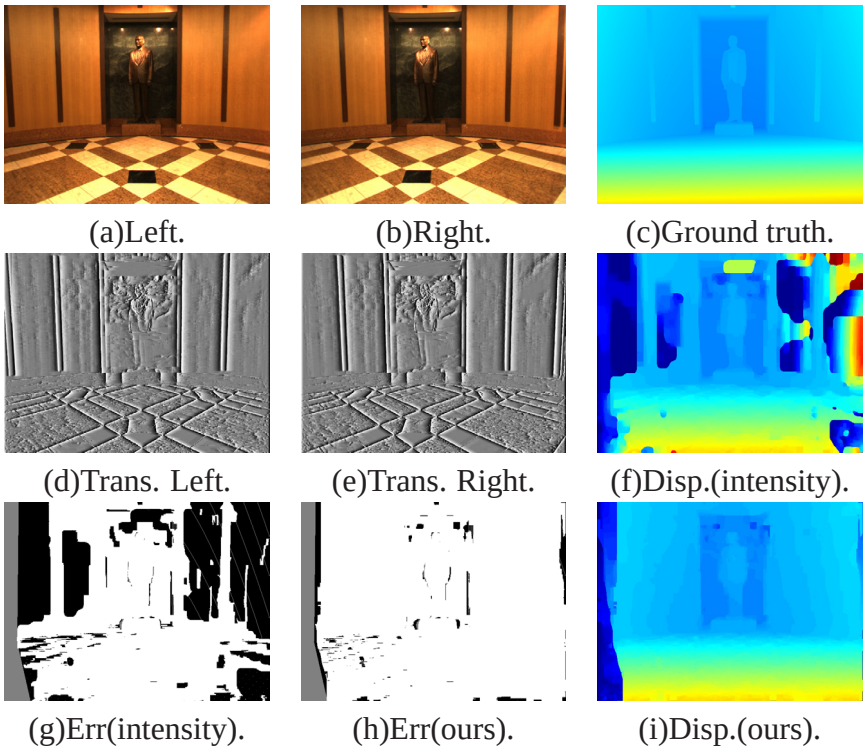


Figure 2.5: Fronto-parallel surfaces.

2.4.3 Visual Evaluation Using Outdoor Scenes

Figure 2.7 presents the experimental results on three outdoor scenes. Figure 2.7 (a) is the reference images from the left lens of the Bumblebee XB3 stereo camera [22]. The images captured by the right lens are omited due to space limit. Panel (b) is the images after transform, (c) is the disparity maps obtained by applying standard BP to the stereo image pairs, (d) is the disparity maps obtained by applying



(a)Left. (b)Right. (c)Ground truth.
(d)Trans. Left. (e)Trans. Right. (f)Disp.(intensity).
(g)Err(intensity). (h)Err(ours). (i)Disp.(ours).

Figure 2.6: Cylindrical surfaces.

standard BP to the images after transform, (e) is the screenshots of the 3D models reconstructed from (c), and (f) is the screenshots of the 3D models reconstructed from (d). The ground-truth disparity maps are not available, but the screenshots of the 3D models presented in Figure 2.7 (f) visually prove that the disparity maps obtained using our method are reasonably accurate. The reconstruction errors due to the lack of texture (white boxes) in Figure 2.7 (c) are successfully removed by our method as shown in (d).

2.4.4 Camera Tracking

In this section, I show that our image transform can be used to estimate the camera motion for low-texture scenes. Our method differs from the standard method in the first step which locates and describes the feature points in each image. Figure 2.8 (a) presents several frames (from the left lens of Bumblebee XB3 stereo camera) of a low-texture wall and the red boxes indicate the feature points detected

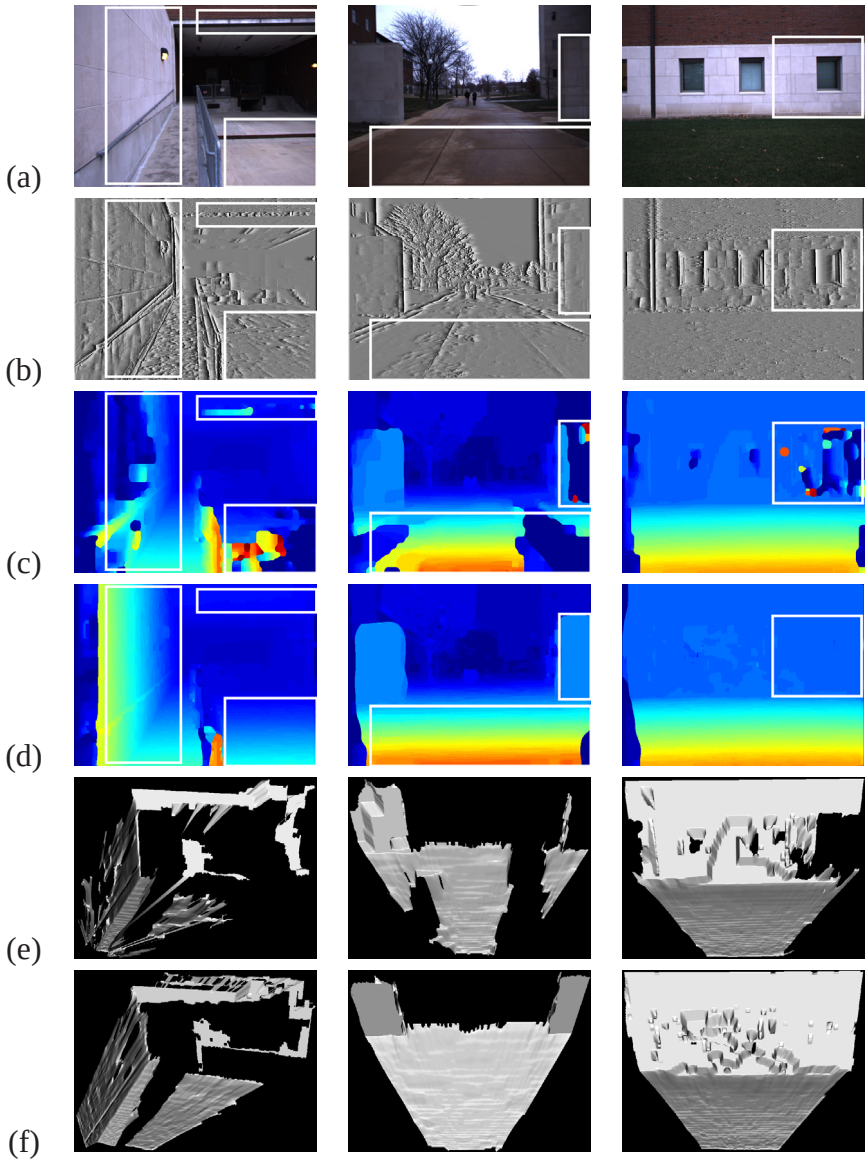
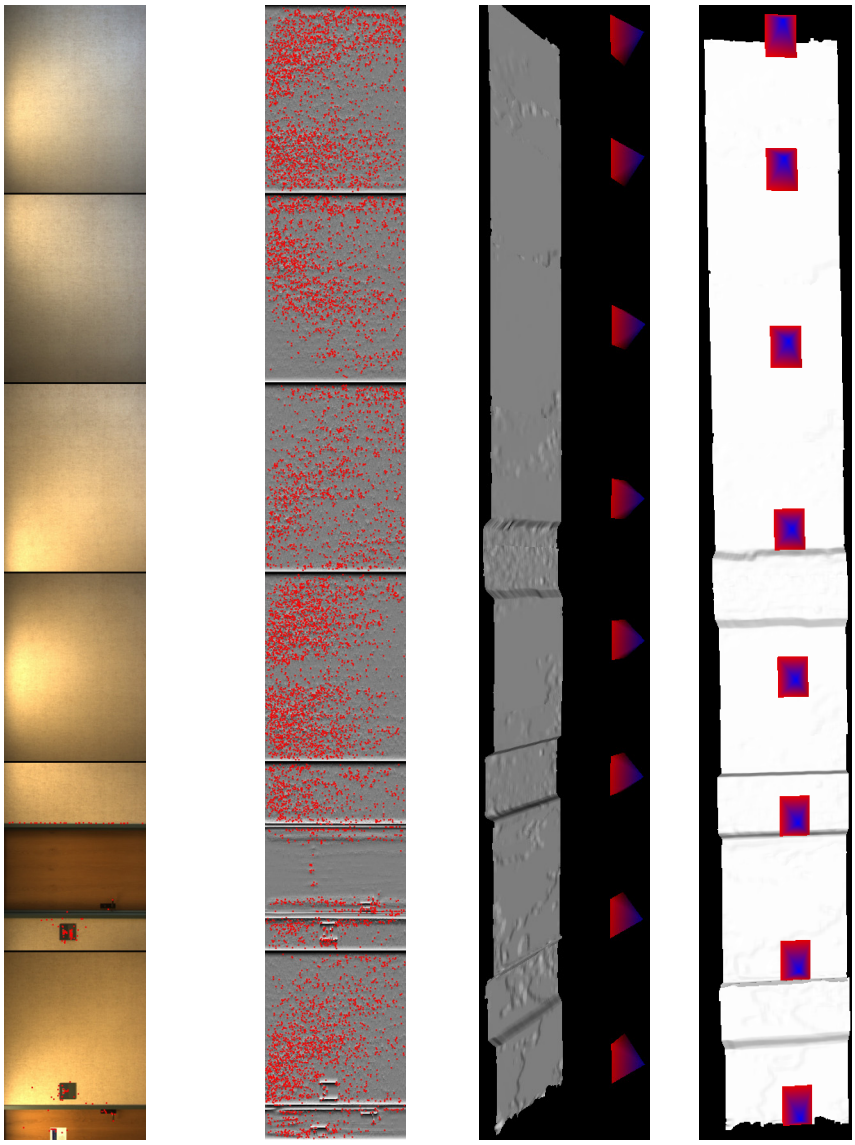


Figure 2.7: Visual evaluation using outdoor scenes.

using SIFT keypoint detector [24].[1] As can be seen in Figure 2.8 (a), no feature point is detected due to the lack of texture in the first few selected frames; it is thus impossible to estimate the camera motion from these frames. Nevertheless, applying SIFT detector to the transformed images in Figure 2.8 (b) shows that many feature points can be detected (red boxes in Figure 2.8 (b)). In addition to the keypoint locations themselves, SIFT provides a local descriptor (computed from the transformed images in Figure 2.8 (b)) for each keypoint. Also, each

[1] I use the demo program provided on the author's website to detect and match the keypoints.



(a)Camera images. (b) Transformed. (c) 3D model. (d) 3D model.

Figure 2.8: Camera tracking.

keypoint has a depth value computed from stereo matching using the transformed images (from the left and right lens of Bumblebee XB3 stereo camera). Next, for every neighboring frame pair, I matched keypoint descriptors between them and converted the matched keypoints into two 3D point clouds using the depth values at each keypoint. I finally estimate the best rotation and translation (in a least squares sense) that transform these two 3D point clouds using the method presented in [25]. Figure 2.8 (c) and (d) present screenshots of the 3D model reconstructed using the estimated camera rotation and translation parameters, which

visually demonstrate that the keypoint detection, description and the depth estimation using the transformed images are accurate. Also, from the reconstructed 3D model, I can calculate the distance between the camera centers of the first and last frame which is 15.5 meters. For quantitative evaluation, I manually measured the distance between the positions where the first frame and the last frame were capture. The measured distance is 15.3 meters, which is close to that estimated using only images.

2.5 Discussion

I have presented a new image transform - epipolar distance transform. The transform is affine invariant, and the transformed image can be directly used for stereo matching. This transform captures the structure of local regions by computing the ratios of lengths along the epipolar lines, which is affine invariant and produces variances inside low-textured regions. Hence, it has the following advantages compared to image intensity:

1. It is robust for matching low-texture regions.
2. It is robust to keypoint detection and description for low-texture scenes.
3. It is robust to lens vignetting.

Besides, the transform is suitable for real-time applications as it can be computed very fast: over 30 frames per second for a VGA-sized image on a DELL XPS laptop computer with a 2.5 GHz Intel Core 2 Duo processor, and over 500 frames per second on an NVIDIA 8800 GTX GPU.

Since the transform is computed based on the geometric properties of the image segments, it is invalid for occluded segments, such as the black pixels in Figure 2.4 (a). Another unsolved problem is how to estimate the parameter σ_S at different pixel location. It is currently set to a constant, but should be determined based on the size of the segments in theory. Nevertheless, the use of integral histogram enables setting σ_S adaptively at each pixel location without extra computation. I intend to investigate these problems in the future.

CHAPTER 3

SIMULTANEOUS ESTIMATION OF ILLUMINATION CHROMATICITY, CORRESPONDENCE AND SPECULAR REFLECTION

3.1 Introduction

A common assumption in many low-level vision problems is that the scene surface is made of Lambertian objects. When dealing with non-Lambertian objects, many problems have to be addressed, e.g., illumination chromaticity estimation [26], [27], [28], specularity removal [29], [30], [31], [32], [33], [34] and correspondence searching for non-Lambertian surfaces [35], [36], [37], [38]. Usually, they are treated separately. Assumptions are made about one factor to solve another.

Many illumination chromaticity estimation methods assume that the specular highlight regions are detected as a pre-processing step. Lee [26] introduced a method of estimating illumination chromaticity using highlights from surface parts with at least two colors, which required segmenting the colors of the highlights. This will cause problems when dealing with heavily textured surfaces. Finlayson and Schaefer [27] extended Lee's algorithm but were still not able to avoid the segmentation problem. Tan et al. [28] proposed an illumination chromaticity estimation approach for single/multi-colored surfaces without using color segmentation, but still required the detection of rough highlight regions achieved by setting a heuristic threshold on the image intensity. Chakrabarti et al. [39] proposed a method without detecting the highlights, but it requires building an explicit statistical model learned from images collected under a known illuminant.

With the pre-knowledge of the illumination chromaticity, many specular highlight removal methods can be carried out. For instance, Lin and Shum [29] changed the light source direction to capture two color images with a stable camera and then, by assuming that at least one of the pixels in the two images was diffuse, the diffuse component could be extracted. With a single image which was normalized using the estimated illumination chromaticity, Tan and Ikeuchi used either neighbor-based method [31] or color space analyzing method [30] to

recover the diffuse components. The neighbor-based methods examine the expected diffuse colors of neighboring pixels in the image. These methods require repetitive texture or simply work with low-textured surfaces. Color space analyzing methods analyze the distributions of image colors within a color space; they can be greatly impaired by clutter in the color space that may be caused by a number of factors, including image noise and color blending at edges. Tan et al. [32] later presented a method taking advantage of both kinds of methods. An SUV color space was proposed by Mallick et al. [33] which separated the specular and diffuse components into S channel and UV channels respectively based on the knowledge of the illumination chromaticity. Mallick et al. [34] also used the SUV space for highlight removal by iteratively eroding the specular channel using either a single image or video sequences.

Using the estimated specular-free images or the detected highlight regions, researchers developed different correspondence searching methods for non-Lambertian surfaces. Zickler [35] further explored the SUV color space proposed in [33]. By taking the ratio of the diffuse channels, a new specular invariant was extracted, and both the diffuse channels and the invariant were then used for stereo matching, optical flow, shape from shading and photometric stereo. Yoon and Kweon [36] extracted a two-band specular-free image for correspondence matching. Correspondence searching can also be performed by rejecting the detected specular regions as outliers. In [40], specular pixels in multi-view images were detected first by computing the uncertainty of depth estimates. Detected pixels were then treated as outliers when computing the similarity among pixels to reduce the effect of specular reflection. Bhat and Nayar [41] considered the likelihood of correct stereo matching by analyzing the relationship between stereo vergence and surface roughness, and also proposed a trinocular system where only two images were used at a time in the computation of depth at a point. Blake [37] and Brelstaff and Blake [38] excised specularities as a pre-processing step.

Contributions: In this chapter, I provide a new solution to three vision problems: illumination chromaticity estimation, correspondence searching and specularity removal. The foundation of the solution is a new matching invariant called *Illumination Chromaticity Constancy* introduced in the chapter. I search for correspondence by analyzing the chromaticities of the color differences between the corresponding pixels in two camera images, and define it as *Chromaticity Match*. Each correspondence hypothesis is associated with a Chromaticity Match, and will vote for a specific illumination chromaticity hypothesis according to this match value.

In this chapter I assume that the objects are stationary and the images are captured by a moving camera under the same illuminant. Thus for correct correspondence hypotheses, the corresponding Chromaticity Match values will be the same as the illumination chromaticity which is constant, and the correct illumination chromaticity hypothesis will win with the greatest amount of votes. However, there is a significant amount of noise contributed by (i) a large number of incorrect correspondence vectors, and (ii) because the highlight areas are usually much smaller than the diffuse areas and the latter contribute noise. I then apply the local smoothness constraint which is popularly used in the field of correspondence matching to suppress noise. Unlike previous methods, our illumination chromaticity estimation approach does not require detecting the specular pixels. Also, because Chromaticity Match computed from the correct correspondence will be equal to the estimated illumination chromaticity, it can then be used as a new matching invariant to match highlights. With an additional assumption that the highlights in the two images do not spatially overlap, and thus the diffuse component of a pixel in the highlight can be extracted by finding its corresponding pixel in the other view, the diffuse reflection can be estimated too. Our experimental results demonstrate the effectiveness and robustness of our method as more inputs/constrains are used.

3.2 Illumination Chromaticity Constancy

Surface reflection of dielectric inhomogeneous objects can be described with the dichromatic reflection model [42], which states that the light reflected from an object is a linear combination of diffuse and specular reflections. Let Γ_c ($c \in \{R, G, B\}$) denote the fraction of the color component c present in the illumination, called illumination chromaticity. Let $m_s(p)$ denote the total specular reflection from a pixel p over all colors. Using the dichromatic reflection model, the color values of p in an image I taken by a RGB camera can be represented as

$$I_c(p) = I_c^{diff}(p) + I_c^{spec}(p) = I_c^{diff}(p) + m_s(p)\Gamma_c, \tag{3.1}$$

where $m_s(p) = \sum_{c\in\{r,g,b\}} I_c^{spec}(p)$ and

$$\Gamma_c = \frac{I_c^{spec}(p)}{m_s(p)}. \tag{3.2}$$

Assume I and J are two images captured by the same camera from different viewing directions. Let p be a pixel in image I, and $\bar{p} = p + D(p)$ be its corresponding pixel in J at a relative location given by the correspondence vector $D(p)$. I then define *Chromaticity Match* as

$$M_c(p, D(p)) = \frac{(I_c(p) - J_c(p + D(p)))}{\sum_{c \in \{R,G,B\}} (I_c(p) - J_c(p + D(p)))}. \tag{3.3}$$

$M_c(p, D(p))$ is set to zero when $I_c(p) = J_c(p + D(p))$. If both the objects and the light source stay still, the diffuse components of the correctly matched pixels will be the same, and their color difference will equal the difference of their specular components: $I_c(p) - J_c(p + D(p)) = (m_s(p) - m_s(p + D(p)))\Gamma_c$. As a result, Chromaticity Match of the correctly matched pixels will be the same as the illumination chromaticity Γ_c. I refer to this property as *Illumination Chromaticity Constancy*, and use it to simultaneously estimate pixel correspondences and Γ_c, which is assumed to be constant across the scene. An estimate of Γ_c helps match pixels within specularities, and the knowledge of pixel correspondences helps estimate Γ_c.

3.3 Estimate Illumination Chromaticity via Matching

I first formulate the illumination chromaticity estimation problem as a maximum likelihood (ML) estimation problem which infers the illumination chromaticity given the two input color images and the correspondence vectors. Using Bayes' rule, the optimal solution Γ is given by

$$\underset{\Gamma}{\operatorname{argmax}}\, P(\Gamma | I, J, D) = \underset{\Gamma}{\operatorname{argmax}}\, \frac{P(I, J | \Gamma, D) P(\Gamma, D)}{P(I, J, D)}, \tag{3.4}$$

where Γ is the illumination chromaticity (same for every pixel because the illumination is assumed to be chromatically uniform), D is the correspondence vectors, and I and J are the camera images. Let H_I denote the set of highlights in image I, and p denote a pixel in I. Assumed that the observation noise at different pixels follows an independent identical distribution (i.i.d.),

$$P(I, J | \Gamma, D) \propto \prod_{p \in H_I} exp(-F(I, J, \Gamma, p, D(p))), \tag{3.5}$$

where $D(p)$ represents the correspondence vectors for $p \in H_I$ and $F(I, J, \Gamma, p, D(p))$ is the cost function which measures the difference between the estimated value of the illumination chromaticity Γ and the Chromaticity Match $M(p, D(p))$ estimated from the individual pixel correspondences as defined in Eqn. (3.3), all of which should ideally be the same as Γ (I dropped the subscript c for simplicity). In practice, both i.i.d. and not i.i.d. noise are presented in images. One type of noise that is reasonably well modeled as i.i.d. is sensor noise (e.g., shot noise), and noise that occurs due to certain environmental effects that cause spatial bursts is obviously not i.i.d. Our approach handles only i.i.d. noise. Also, I consider only the low values of F because the others arise from wrong correspondences. To this end, I limit the F value of interest to 1, by using the truncated squared difference as the cost:

$$F(I, J, \Gamma, p, D(p)) = \min(\frac{(\Gamma - M(p, D(p)))^2}{2\sigma_p^2}, 1), \tag{3.6}$$

where $\sigma_p < 1$ is the standard deviation of the pdf of $(I, J|\Gamma, D)$ as in Eqn. (3.5). In this chapter, I set $\sigma_p = 0.01$.

Since D (representing 3D structure) and Γ (representing light source) are independent,

$$P(\Gamma, D) = P(\Gamma)P(D). \tag{3.7}$$

Without any knowledge of the light source, I will assume that $P(\Gamma)$ is uniformly distributed in $[0, 1]$, i.e., $P(\Gamma) = 1$. For the given input images I and J and the given correspondence vectors D, $P(I, J, D) = 1$ and $P(D) = 1$.

Taking the logarithm on both sides of Eqn. (3.4), and using Eqn. (3.5), the optimal Γ is given by

$$\begin{aligned} \underset{\Gamma}{\operatorname{argmax}} \log(P(\Gamma|I, J, D)) &= \underset{\Gamma}{\operatorname{argmax}} \sum_{p \in H_I} (-F(I, J, \Gamma, p, D(p))) \\ &= \underset{\Gamma}{\operatorname{argmax}} \sum_{p \in H_I} (1 - F(I, J, \Gamma, p, D(p))). \end{aligned} \tag{3.8}$$

To obtain the value of Γ, I compute votes for discretized values of Γ. I define the vote distribution $V(\Gamma)$ as

$$V(\Gamma) = \sum_{p \in H_I} V(\Gamma, p, D(p)) = \sum_{p \in H_I} (1 - F(I, J, \Gamma, p, D(p))). \tag{3.9}$$

I select the Γ with the largest vote value as the solution. The vote distribution is computed for each channel separately, and the peak of the vote distributions is accepted as the correct illumination chromaticity.

The values of D have not been constrained at all so far. Because I do not know where the highlights are, I must consider all pixels as candidates. Further, since I do not know where the pixel correspondences are either, I must consider all possible pairs. If images I and J are taken from calibrated cameras, I can restrict the correspondences of a pixel in I to lie along its epipolar line in J; namely, I can rewrite Eqn. (3.9) as

$$V(\Gamma) = \sum_{p \in I} \sum_{D(p) \in E(p)} (1 - F(I, J, \Gamma, p, D(p))), \tag{3.10}$$

where $E(p)$ denotes the epipolar line of p. Alternatively, if the cameras are not calibrated, I will need to search across all possible values through the maximum possible.

In our experiments (Section 3.5), I searched for a pixel's correspondence within a 300×300 window. Thus there is a significant amount of noise contributed by (i) a large number of incorrect correspondence vectors, and (ii) because the highlight areas are usually much smaller than the diffuse areas and the latter contribute noise.

To suppress this noise, a local smoothness assumption (popularly used in the field of correspondence matching) is applied, which enforces similarity between chromaticity estimates at a pixel and those at other pixels within a surrounding window, called the *support* window. Let $N(p)$ be the support window for pixel p in image I and $q \in N(p)$ be corresponding pixels in the support windows $N(p)$; then the local smoothness assumption implies that Chromaticity Match $M(p, D(p)) = M(q, D(p))$ if the correspondence vector $D(p)$ is correct. Using this constraint, I rewrite the vote distribution in Eqn. (3.10) as

$$V(\Gamma) = \sum_{p \in I} \sum_{D(p) \in E(p)} C_p (1 - F(I, J, \Gamma, p, D(p))), \tag{3.11}$$

where

$$C_p = \prod_{q \in N(p)} exp(-\frac{(M(p, D(p)) - M(q, D(p)))^2}{2\sigma_p^2}) \tag{3.12}$$

measures the consistency of the Chromaticity Matchers inside the support win-

dow.

There are other constraints that can be used to suppress the noise. For instance, (i) the three channels of the computed Chromaticity Match should be all positive or negative because the illumination chromaticity will never be negative, and (2) in Eqn. (3.3), if $|I_c(p) - J_c(p + D(p))|$ is smaller than a threshold (set to 10), the corresponding Chromaticity Match will be dropped to avoid quantization noise.

Note that although our framework is based on binocular matching, it can be easily extended for multiview matching. One simply performs binocular matching and accumulates the vote distribution on every two images.

3.4 Stereo Matching and Highlight Removal

In stereo vision, the corresponding pixels in the stereo images are assumed to have the same texture/color values, and in this chapter, I define this constancy as *texture constancy* (TC). More specifically, let I and J be the left and right camera images, p be a pixel in image I and $\bar{p} = p + D(p)$ be its corresponding pixel in J at a relative location given by the correspondence vector $D(p)$, $N(p)$ be the support window for pixel p and $q \in N(p)$ be the pixels inside the support window $N(p)$, the dissimilarity between pixel p and $\bar{p}$ is then measured by aggregating raw matching costs in the support window $N(p)$:

$$E(p, D(p)) = \sum_{q \in N(p)} \sum_{c \in \{R,G,B\}} |I_c(q) - J_c(q + D(p))|. \tag{3.13}$$

The correspondence hypotheses ($D(p)$) corresponding to the smallest dissimilarity values ($E(p, D(p))$) are accepted as correct. For a rectified stereo image pair, the corresponding pixels lie in the same horizontal scanline, and the shifted amount is called the disparity in stereo vision. The correspondence vectors are then represented as a 2D image called disparity map. The disparity value (z) and the depth value (d) of a pixel are related by the product of the focal length (f) and the baseline (b) between the two cameras:

$$z \cdot d = f \cdot b. \tag{3.14}$$

TC is invalid for specular pixels because the highlight shifts as the camera moves; that is, $I_c(p) \neq J_c(p + D(p))$ when either pixel p in image I or its corre-

spondence $p + D(p)$ in image J is specular. Nevertheless, Section 3.3 shows that for specular pixels, Chromaticity Match computed from the correct correspondence is the same as the estimated illumination chromaticity: $M_c(p, D(p)) = \Gamma_c$. This property is defined as *Illumination Chromaticity Constancy* (ICC) in Section 3.2 and can be used to match pixels inside specular highlights. However, ICC is invalid for diffuse pixels.

TC and ICC can be integrated in some manner to match both diffuse and specular pixels. Letting

$$H(q, D(p)) = \sqrt{\frac{1}{3} \sum_{c \in \{R,G,B\}} (M_c(q, D(p)) - \Gamma_c)^2}, \tag{3.15}$$

the integration is obtained by re-defining the dissimilarity in Eqn. (3.13) as

$$E(p, D(p)) = \sum_{q \in N(p)} H(q, D(p)) \cdot \sum_{c \in \{R,G,B\}} |I_c(q) - J_c(q + D(p))|, \tag{3.16}$$

when the three channels of $M_c(q, D(p))$ are all positive or all negative and $H(q, D(p)) > 0.1$ (to exclude pixels that have high probability of being diffuse).

Finally, I relate the correct correspondences to the diffuse components of the specular pixels by assuming that the highlights in the two images do not spatially overlap; thus the diffuse component of a pixel in the highlight can be extracted by finding its corresponding diffuse pixel in the other view.

One problem associated with our method is that it is invalid for specular pixels that are saturated. Additionally, the stereo reconstruction quality of our method decreases as the areas of the overlapping highlights increase, since the color differences of the specular pixels and their correspondences inside the overlapping highlights will be small and the precision of Chromaticity Match ($M_c(p, D(p))$, Eqn. 3.3) will decrease due to quantization (8-bit images are used in our experiments). But note that our stereo matching method does not require the correspondence of a specular pixel to be diffuse. In fact, our method requires that their color difference should be large enough to avoid the quantization noise arising from the computation of Chromaticity Match. However, if the color difference is very small, TC will be valid for both specular and diffuse pixels.

These difficulties can be greatly reduced if images captured from many view directions/positions are available. From these images, coarse estimates can be obtained from each selected stereo pair, and then fused to give coherent estimates. In

this chapter, the coarse depth maps are fused using the efficient method presented in [43]. The fused depth maps are then used to remove highlights. For each pixel p in each image, I project it to the other images using its depth value to obtain the colors of its correspondences $\bar{p}_i$ in the other images. If the luminance of $\bar{p}_i$ is much larger than the luminance of p, that is, the difference is larger than a constant (set to 10 in our experiment), then $\bar{p}_i$ is treated as a specular pixel. The median value of the colors of all the correspondences $\bar{p}_i$ (each color band is processed separately) which are believed to be diffuse are then selected as the correct diffuse reflection of pixel p. The median values are used such that the diffuse reflections are consistent when viewed from different directions/locations.

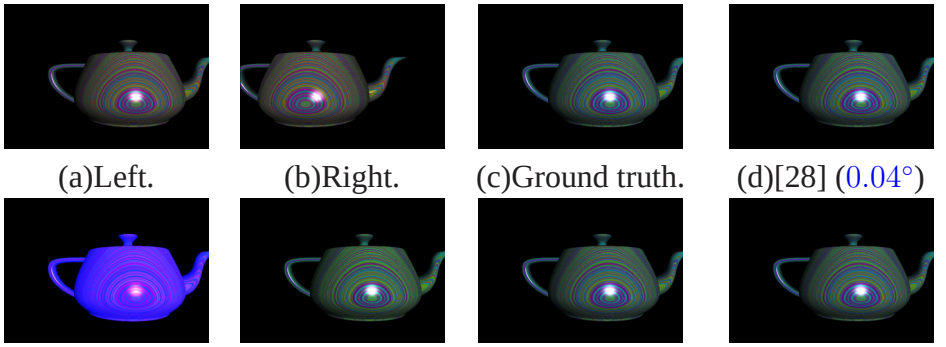
3.5 Experimental Results

To evaluate our method, I conducted experiments on a synthetic data set and several real images captured by a Sony DFW-X700 camera with gamma correction turned off. To quantitatively evaluate the proposed illumination chromaticity estimation method for real scenes, I compared the results with the average value of the image chromaticity of a white reference image captured by the same camera. Specifically, I cast light on white printing papers and then capture it using the same camera under the same setting. I also compared our results with the methods presented in [28] and [39]. For highlight removal, I compared our results with images captured with polarizing filters over the camera and the light source. Comparison with the highlight removal method presented in [31] is also provided. Because [31] assumes that the illumination chromaticity is known, in our experiments, the ground-truth illumination chromaticity (measured with a white reference) is used.

3.5.1 Synthetic Data Set

Figure 3.1 visually compares our illumination chromaticity estimation method with [28] and [39]. It is apparent that the method presented in [39] is not suitable for this synthetic data set as Figure 3.1 (e) is not comparable to the ground truth in Figure 3.1 (c). The blue angle numbers in the brackets under (d)-(h) are the angular error of estimated illumination chromaticities, which numerically prove that with a 5×5 support window, our method can obtain the most accurate estimates and that [39] is invalid for this data set. Figure 3.2 presents the corresponding vote

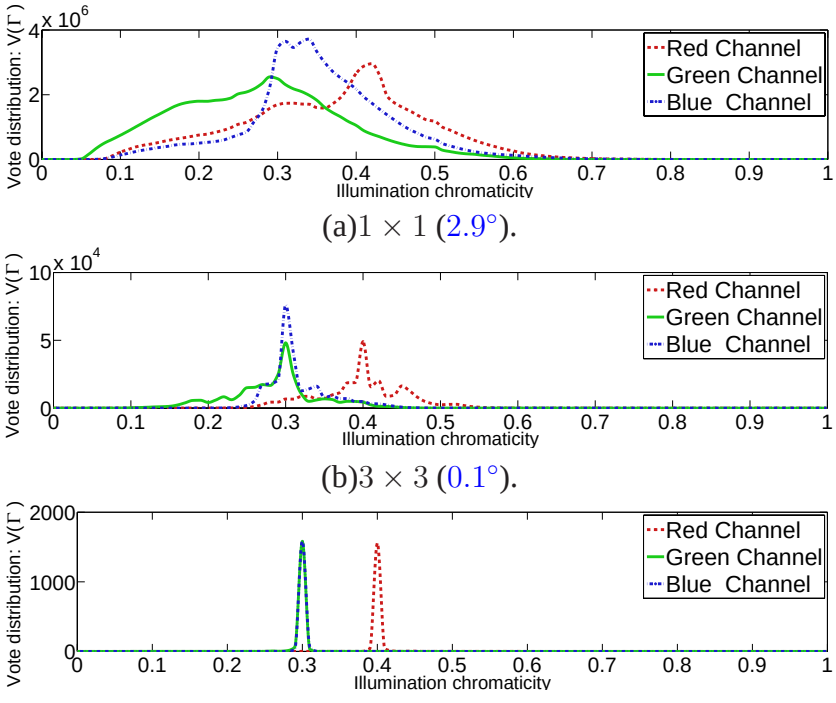
distributions for our method with three support window sizes: 1×1, 3×3 and 5×5. Note that the vote values decrease dramatically from window size 1×1 to 3×3 due to resulting ability to identify inconsistent correspondences (with dissimilar local Chromaticity Matches).



(a)Left. (b)Right. (c)Ground truth. (d)[28] ($0.04°$)

(e)[39] ($34.9°$). (f)1×1 ($2.9°$). (g)3×3 ($0.1°$). (h)5×5 ($0.007°$).

Figure 3.1: Illumination chromaticity estimation for the synthetic images.

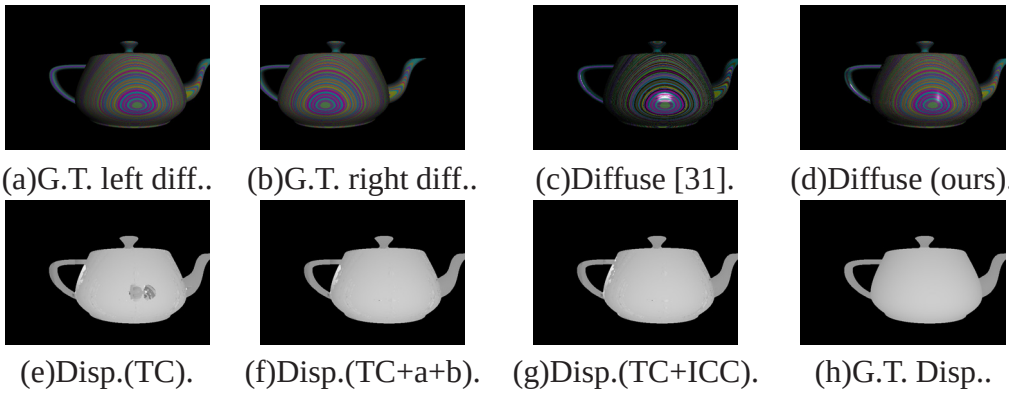


(a)1×1 ($2.9°$).

(b)3×3 ($0.1°$).

(c)5×5 ($0.007°$).

Figure 3.2: Illumination chromaticity vote distributions for the synthetic images in Figure 3.1.



(a)G.T. left diff.. (b)G.T. right diff.. (c)Diffuse [31]. (d)Diffuse (ours).

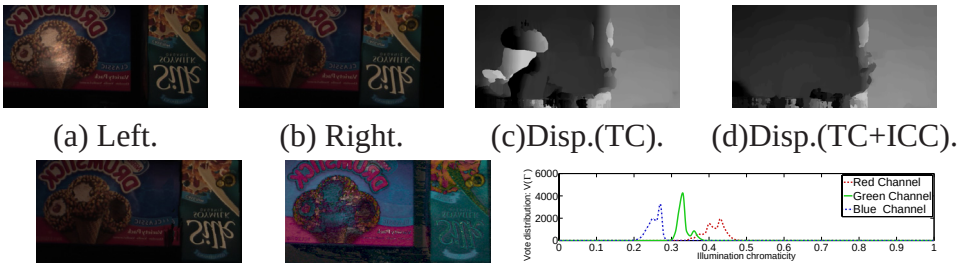
(e)Disp.(TC). (f)Disp.(TC+a+b). (g)Disp.(TC+ICC). (h)G.T. Disp..

Figure 3.3: Stereo matching and highlight removal for the synthetic images.

Figure 3.3 presents the results for the depth estimation and highlight removal methods using the stereo pair presented in either Figure 3.1 (a)-(b) or Figure 3.3 (a)-(b). Figure 3.3 (a) and (b) are the ground-truth diffuse reflections, while (c) and (d) are the estimated diffuse reflections using the method presented in [31] and our method. The disparity map obtained using our method (Figure 3.3 (g)) is used to compute the diffuse reflection in (d). As can be seen, [31] is invalid for this data set as the highlights in (c) are not removed. Also, the colors in (c) are different from the ground truth in (a). Our results is better than [31] as can be seen in (d). However, some of the specularities are not removed due to the violation of the assumption that the highlights in the two images should not spatially overlap. Nevertheless, our stereo matching method is robust to the violation of this assumption as can be seen in (g). Visual comparison with the ground-truth disparity map in (h) shows that the estimated disparity values of the specular pixels are correct in (g). Image (e) is the disparity map obtained from standard stereo matching method (using TC). The disparity map in (f) is also estimated using standard method, but the input images are free of specularity, which means that the ground-truth diffuse reflections in (a) and (b) are used as input. Figure 3.3 (e)-(f) show that this standard method is invalid for specular highlights. Image (g) is the disparity map estimated from the proposed stereo matching method which integrates TC and ICC. Note that (g) is visually very similar to (f). Let a pixel be a bad pixel if the difference between the estimated disparity value and the ground truth is larger than 1 [44]; the percentages of bad pixels in (e)-(g) are 5.41%, 4.29% and 4.26%, which numerically prove that our method is suitable for matching specular highlights.

3.5.2 Real Data Sets

I next present our experimental results on real data sets. Figure 3.4 provides the experimental results with a real stereo image pair. The illumination chromaticity vote distribution with a 5×5 support window is presented in Figure 3.4 (g), and the angular errors are $2.20°$ (blue angle number in the bracket under (g)). The estimated disparity map presented in (d) shows that our method is able to remove the matching errors in (c) due to specular highlights. The estimated diffuse reflections presented in (e) and (f) show that our highlight removal method is more robust than the single view based highlight removal method presented in [31] as the highlights in (f) are not removed. Also, Figure 3.4 (f) shows that [31] is invalid for neutral pixels, and the estimated diffuse colors are incorrect. However, our highlight removal method obtained incorrect diffuse colors around the half-occluded regions due to the incorrect correspondences. This problem can be greatly reduced when more images are used. Figures 3.5, 3.6 and 3.7 present the experimental results with multiple images. Specifically, Figure 3.5 and 3.6 used 21 images, and Figure 3.7 used 15 images.



(a) Left. (b) Right. (c)Disp.(TC). (d)Disp.(TC+ICC).

(e) Diffuse (ours). (f) Diffuse [31]. (g)5×5 ($2.20°$).

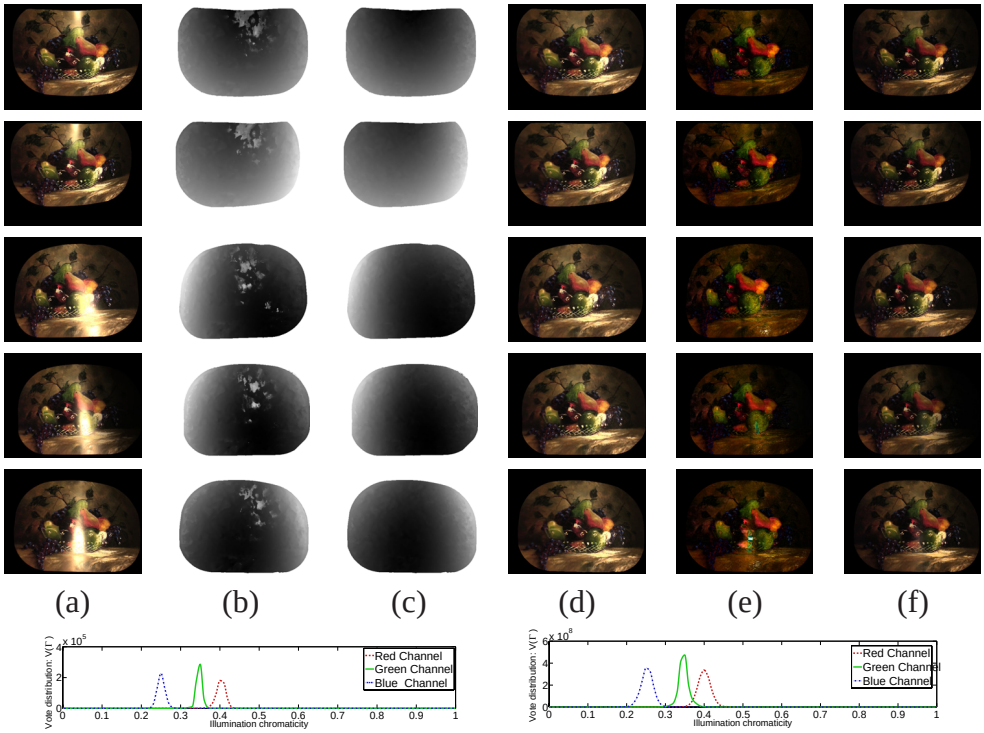
Figure 3.4: Binocular matching.

The angular errors of the estimated illumination chromaticity of the three real data sets (Figure 3.5, 3.6 and 3.7) using only two of the input images are $2.27°$, $2.81°$ and $12.97°$. The last data set (Figure 3.7) has large angular error as the highlights are very sparse and most of them are saturated. However, using more images can reduce the estimation error. The angular errors of the three real data sets using five of the input images (presented in Figure 3.5 (a), Figure 3.6 (a) and Figure 3.7 (a)) are $1.97°$, $1.24°$ and $5.74°$. As can be seen, the estimation error drops with respect to the increasing number of input images.

Visual comparison of the depth maps presented in Fig 3.5 (b)-(c) and Fig 3.6 (b)-(c) shows that our stereo matching can greatly improve the reconstruction ac-

curacy for specular pixels. However, since the specular highlights in Figure 3.7 (a) are very sparse, the depth map estimated using our method (Figure 3.7 (c)) is only slightly better than the standard method (Figure 3.7 (b)).

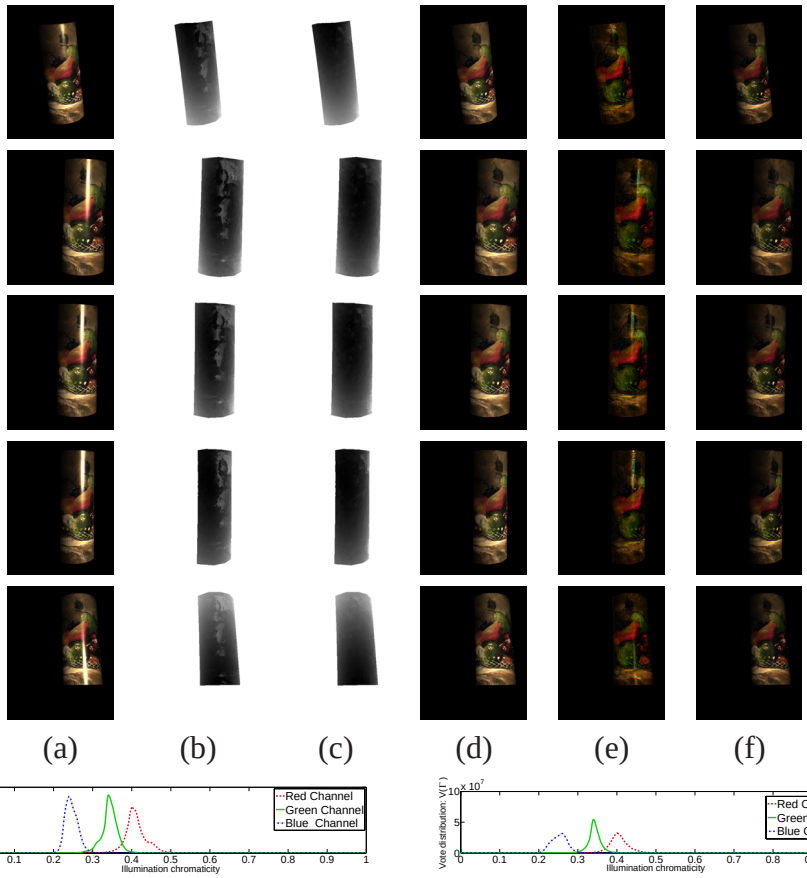
Figures 3.5 (d)-(f), Figures 3.6 (d)-(f) and Figures 3.7 (d)-(f) visually compare the estimated diffuse reflections using our method, the method presented in [31] and the ground truth. Visual comparison shows that our method outperforms [31] as the colors of estimated diffuse reflections using [31] are incorrect (see Figure 3.5 (e), Figure 3.6 (e) and Figure 3.7 (e)) and [31] is invalid for saturated pixels.



(g)5×5 (using two images, $2.27°$). (h)5×5 (using five images, $1.97°$).

Figure 3.5: Multiview matching.

To evaluate the robustness of the proposed illumination chromaticity estimation method, I conducted experiments on a total of 17 real data sets. Besides the four data sets presented in Figures 3.4, 3.5, 3.6 and 3.7, another 13 real data sets are provided in Figures 3.8 and 3.9. The illumination chromaticities of the real illuminants were grouped into five different sets using the image chromaticities of the white reference: $\{0.13158, 0.40248, 0.46829\}$, $\{0.38781, 0.32666, 0.28822\}$, $\{0.44792, 0.31771, 0.23437\}$, $\{0.61730, 0.36693, 0.01197\}$ and $\{0.40070, 0.33177, 0.26676\}$. I calculated the estimation errors by comparing the chromaticity esti-



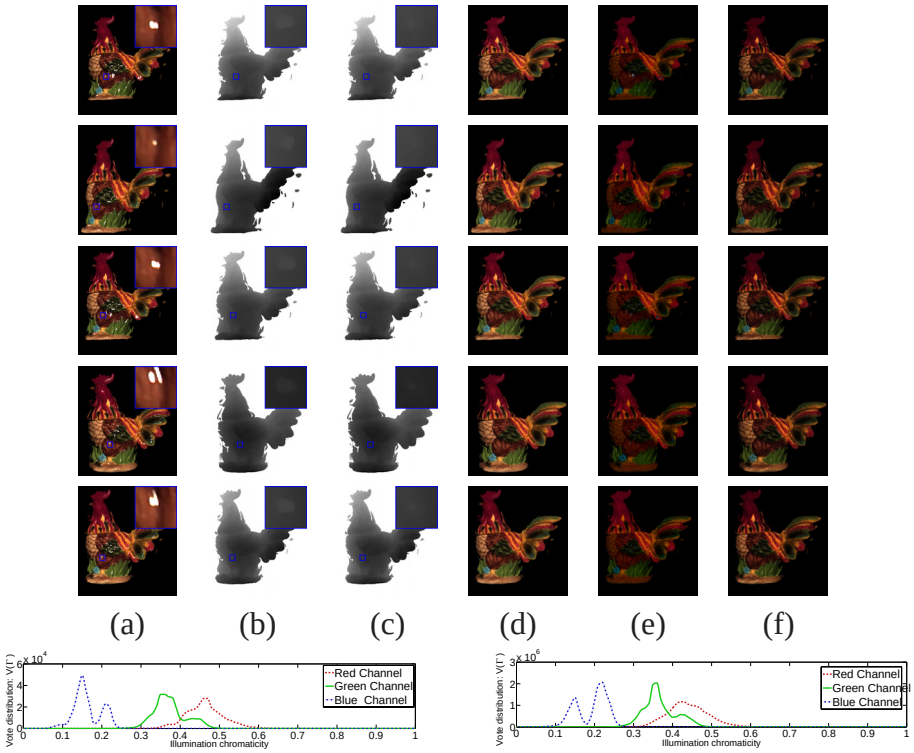
(g)5×5 (using two images, 2.81°). (h)5×5 (using five images, 1.24°).

Figure 3.6: Multiview matching.

mates with those of the white reference. The angular errors are shown in Figure 3.10, and a summary of the experimental results is provided in Table 3.1. The results from the methods presented in [28] and [39] are also included. The error rates demonstrate that our method is generally robust, and our method appears better than [28] and [39] on average for these data sets.

Table 3.1: Summary of the quantitative evaluation of the proposed illumination chromaticity estimation approach for real data sets.

Algorithm	Mean angular error	Minimum angular error	Maximum angular error	Std. dev. of angular error
Ours	1.98°	0.57°	5.74°	1.34°
[28]	7.31°	0.84°	36.8°	8.64°
[39]	7.27°	0.16°	27.9°	7.23°



(g)5×5 (using two images, $12.97°$). (h)5×5 (using five images, $5.74°$).

Figure 3.7: Multiview matching.

3.6 Discussion

A new invariant called *Illumination Chromaticity Constancy* for matching highlights between images is introduced in the chapter. An algorithm is presented that uses this invariant for three vision problems: illumination chromaticity estimation, correspondence searching and specularity removal. In relation to previous approaches, the most significant advantage of the presented method is that I have related the correct correspondence vectors to both the illumination chromaticity and the diffuse components of the specular pixels, and have presented an attempt to estimate these properties in a uniform framework. Additionally, our method does not require detecting the specular highlights. However, if the illuminant and the object surface have the same chromaticity, the proposed correspondence matching and highlight removal method will fail as every possible Chromaticity Match will be equal to the illumination chromaticity. Nevertheless, the proposed illumination chromaticity estimation method has no problem under this condition. Also, our framework assumes chromatic surfaces and is invalid for grayscale objects.

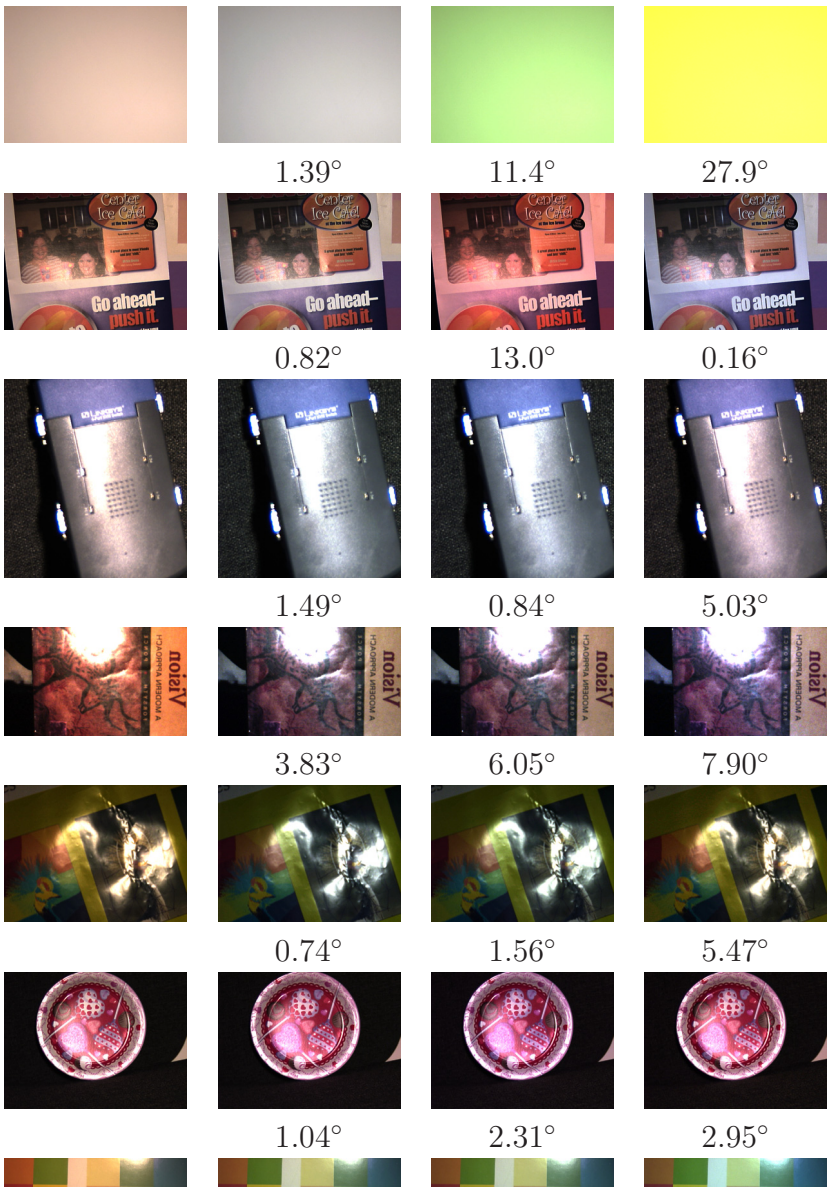


Figure 3.8: Illumination chromaticity estimation for real scenes with corresponding angular errors.

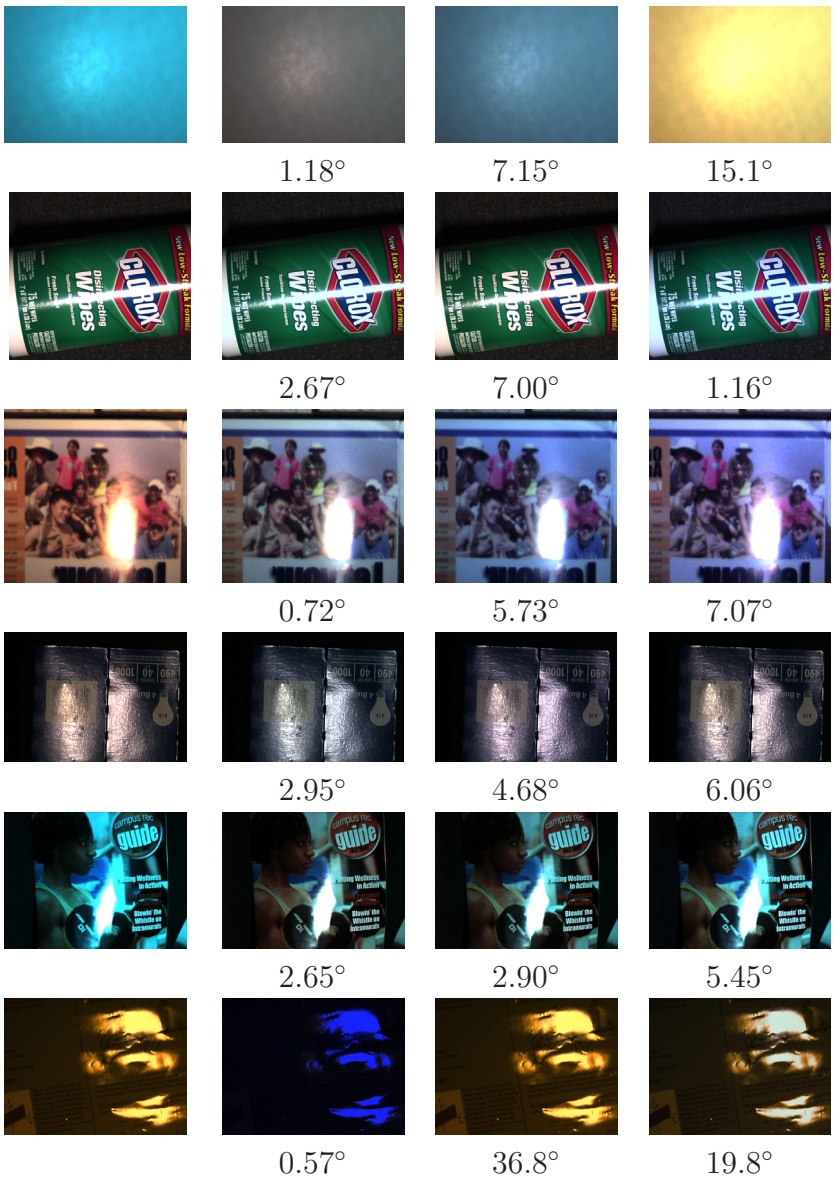


Figure 3.9: Illumination chromaticity estimation for real scenes with corresponding angular errors.

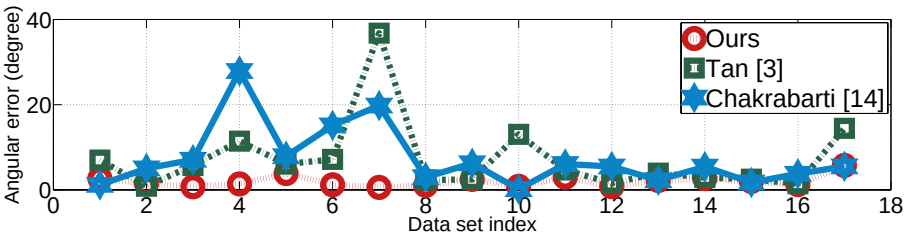


Figure 3.10: Quantitative comparison of the illumination chromaticity estimation methods.

CHAPTER 4

INTEGRATED ESTIMATION OF SURFACE REFLECTANCE AND DEPTH FROM MULTIPLE VIEWS

4.1 Introduction

This chapter is concerned with the estimation of 3D scene properties of surface reflectance and surface shape. For the most part, estimation of each of these properties has been traditionally treated as a separate research topic, e.g., illumination [28, 45], reflection components separation [30, 31, 32], shape from shading [46], and depth from stereo [44, 47, 48].

Recently, a number of methods have been proposed for joint estimation of reflectance and shape. Jin et al. [49] use the rank constraint of radiance tensor as the discrepancy measure to search for correspondences. Once the surface is found, an estimate of the radiance profile for generating view-dependent radiance maps (VDRM) can be computed. Assuming single-material objects, Yu et al. [50] iteratively estimate the shape and a view-independent reflectance map (VIRM) by minimizing the cost of matching input images and the images synthesized using shape and reflectance estimates. With the additional assumption that the illumination direction is known or can be calibrated/measured, Yu et al. [51] propose a method for simultaneous estimation of 3D shape and the Phong reflectance model [52]. Methods presented in [50] and [51] are based on shape-from-shading for objects made of a single material. Also, both Jin and Yu's algorithms are object/scene centered [53], which is less suitable for large scale scenes than image centered [54]. Assuming that the scene points having specular reflection exhibit purely diffuse reflection in some other views, Lin [40] proposes an image centered method to detect specular pixels and estimate their diffuse reflections and depths from other views. However, this method requires that the specularities be sparse.

In this chapter, I propose a novel, single-image based highlight removal method (Section 4.3.1), which is essentially a bilateral filtering process (with the filter size the same as the input image). I will refer to this method as Bilateral Filtering

based Highlight Removal (BFHR) in the chapter. Being $O(1)$ computation, bilateral filtering [55, 56] makes our method computationally efficient. Both visual and quantitative evaluations show that BFHR is more accurate than the propagation based highlight removal (PHR) method proposed by Tan [57], which has the problem of non-convergence and also extracts inaccurate diffuse reflection. I next present a method for simultaneous estimation of depth and reflection (Section 4.3.2). The individual components (stereo matching [58], depth map fusion [43, 59] and diffuse reflection fusion) of the approach are mostly not novel; however, I demonstrate that the integration of all of them through the iterative scheme and the incorporation of confidences in reflection estimates can effectively reduce the reflection separation errors due to saturation, and depth estimation error due to incorrect reflectance estimates. I show that accuracy of both the depth and reflectance estimation increases with respect to the number of iterations (increasing integration), and generally converges in two iterations. I believe that in a mature area, even minor changes which lead to obvious improvement are welcome additions to the field's body of knowledge.

4.2 Overview of the Approach

The algorithm I present can be partitioned into two interacting modules as shown in Figure 4.1. Module 1 is reflection components separation from single view. Module 2 is depth and diffuse reflection estimation from multiple views. These two modules are included as a single iterative optimization procedure (connected by red arrows). In the first module, a set of images representing the diffuse reflection ($\vec{I}_d^{(i-1)}$) and confidence maps ($\vec{C}^{(i-1)}$) are computed from the input images $\vec{I}$. Any single image based highlight removal method can be used in this module. For instance, one can use the method presented in [31] which separates the diffuse and specular components by iteratively propagating the chromaticity information from diffuse pixels to specular pixels. In this chapter, I present a new highlight removal method based on bilateral filtering (BFHR). The second module estimates the diffuse reflection and the depth maps using multi-view stereo matching and depth map fusion. I use the fusion method presented by Nister [59], which was later proved to be computationally efficient by Merrell [43]. At the end of each iteration i, the computed depth maps $\vec{D}_f^{(i)}$ are used to find the corresponding diffuse colors from neighboring diffuse images ($\vec{I}_d^{(i-1)}$). The median values are accepted

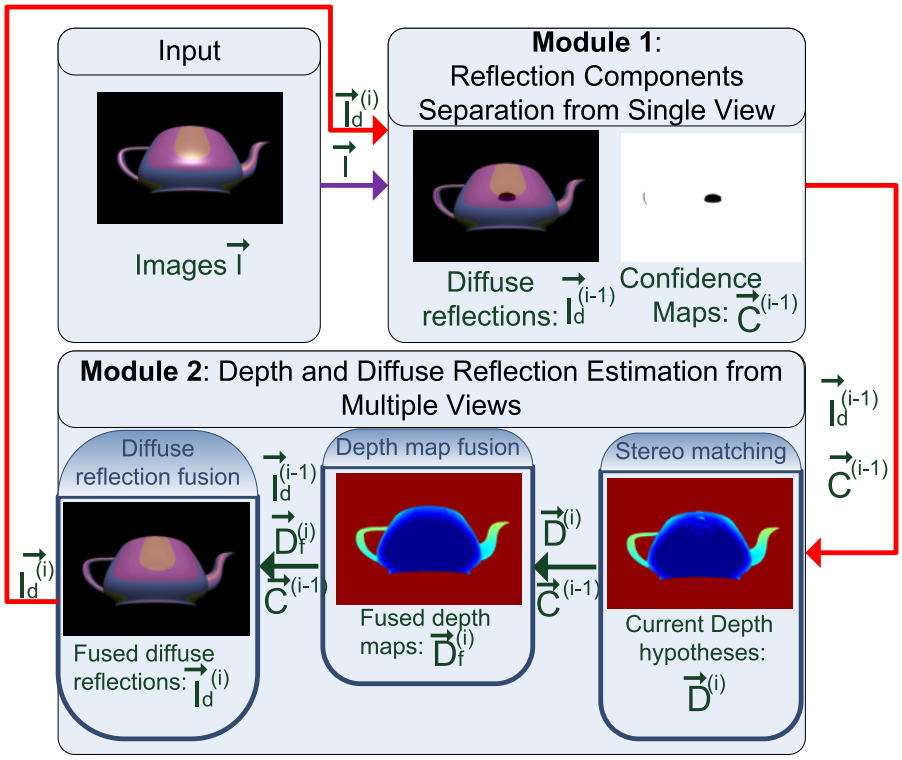


Figure 4.1: The proposed framework.

as the estimates ($\vec{I}_d^{(i)}$). Note that in the first iteration, the colors of saturated pixels (identified using $\vec{C}^{(0)}$) are not taken into account.

4.3 Algorithm

In this section, I give a more detailed description of the modules outlined in Section 4.2.

4.3.1 Bilateral Filter-Based Highlight Removal (BFHR) Using a Single RGB Image

Using standard diffuse+specular reflection models commonly used in computer graphics, the reflected light color ($\vec{J}$) captured by an RGB camera can be represented as a linear combination of diffuse ($\vec{J^D}$) and specular ($\vec{J^S}$) colors:

$$\vec{J} = \vec{J^D} + \vec{J^S}. \tag{4.1}$$

Let chromaticity be defined as the fraction of color component c

$$\sigma_c = \frac{J_c}{\sum_{c\in\{r,g,b\}} J_c}, \tag{4.2}$$

where $c \in \{r, g, b\}$. I define diffuse chromaticity Λ_c and illumination chromaticity Γ_c as follows:

$$\Lambda_c = \frac{J_c^D}{\sum_{c\in\{r,g,b\}} J_c^D}, \tag{4.3}$$

$$\Gamma_c = \frac{J_c^S}{\sum_{c\in\{r,g,b\}} J_c^S}. \tag{4.4}$$

Following the chromaticity definition in Eqns. (4.2), (4.3) and (4.4), I express the reflected light color J_c as

$$J_c = \Lambda_c \sum_{u\in\{r,g,b\}} J_u^D + \Gamma_c \sum_{u\in\{r,g,b\}} J_u^S. \tag{4.5}$$

Assuming that the illumination chromaticity can be measured (with a white reference) or estimated [28], the input image can be normalized such that $\Gamma_r = \Gamma_g = \Gamma_b = 1/3$ and $J_r^S = J_g^S = J_b^S = J^S$. Then the diffuse component can be written as

$$J_c^D = J_c - J^S, \tag{4.6}$$

according to Eqn. (4.5).

Following the chromaticity definition in Eqns. (4.2) and (4.3), I define maximum chromaticity as

$$\sigma_{max} = \max(\sigma_r, \sigma_g, \sigma_b) \tag{4.7}$$

and maximum diffuse chromaticity as

$$\Lambda_{max} = \max(\Lambda_r, \Lambda_g, \Lambda_b). \tag{4.8}$$

Tan [31] shows that the diffuse component can be represented as a function of Λ_{max}

$$J_c^D(\Lambda_{max}) = J_c - \frac{\max_{u\in\{r,g,b\}} J_u - \Lambda_{max} \sum_{u\in\{r,g,b\}} J_u}{1 - 3\Lambda_{max}}. \tag{4.9}$$

Since surface materials may vary from point to point, Λ_{max} changes from pixel to pixel in real images but is limited from $\frac{1}{3}$ to 1.

Estimating the maximum diffuse chromaticity Λ_{max} for every pixel from a single image is a non-trivial problem. However, if it is set to a constant, then a "pseudo-coded" diffuse image which has exactly the same geometrical profile as the diffuse component of the input image can be obtained. In this case, the saturation values of all pixels are made constant and this "pseudo-coded" diffuse image is essentially a 2D image, while the ground-truth diffuse image is a 3D image.

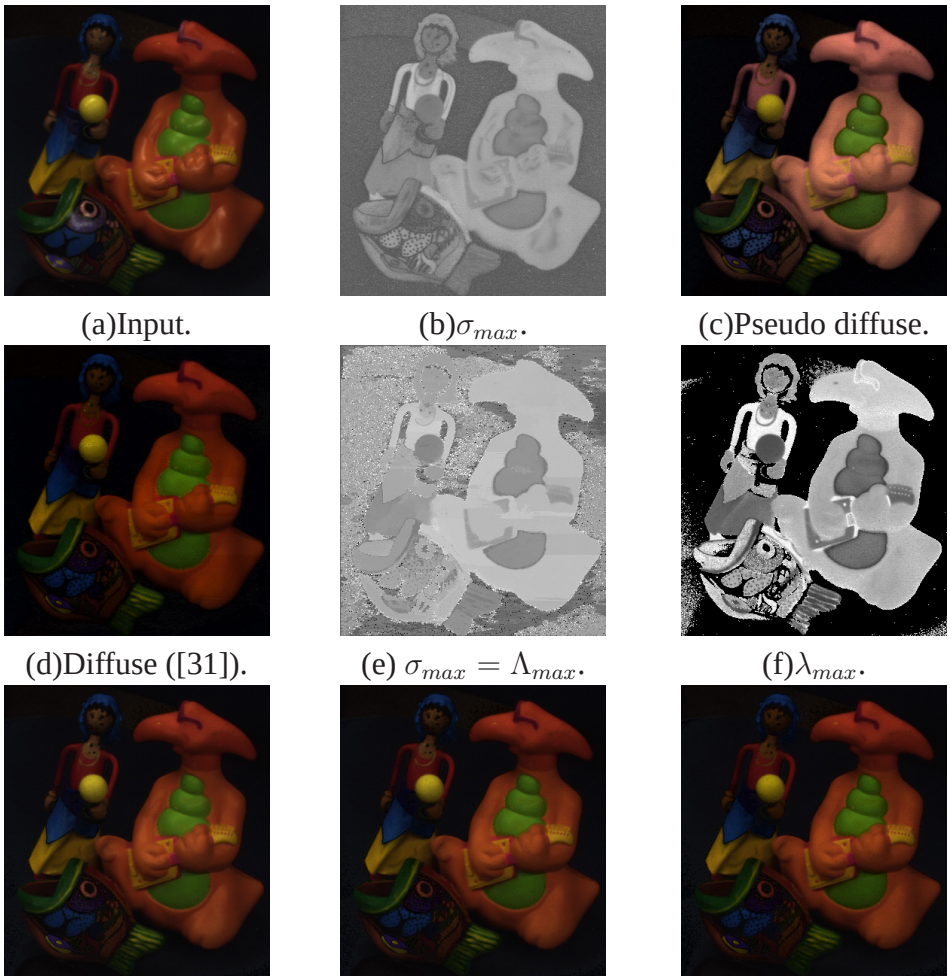
The 2D "pseudo-coded" diffuse image is just an approximation of ground truth, which will fail to preserve the feature discriminability for surfaces having the same hue but different saturation. However, it is the best estimate I can get, and has been demonstrated to be effective for solving the highlight removal problem in [31]. Figure 4.2 presents such an example by setting Λ_{max} to a constant 0.5. Figure 4.2 (a) is the input image, (b) is the maximum chromaticity values σ_{max} computed from the input image (a), (c) is the "pseudo-coded" diffuse image, (d) presents diffuse reflection extracted using the method presented in [31] and (e) presents the maximum chromaticity values σ_{max} computed from (d) using Eqns. (4.2) and (4.7). Assuming that the specular highlights are correctly removed from (d), (e) is also presented as the maximum diffuse chromaticity Λ_{max}.

According to Eqn. (4.9), the highlight removal problem can be reduced to searching for the maximum diffuse chromaticity Λ_{max} which changes from pixel to pixel. However, as shown in Figure 4.2 (d) and (e), the variance of Λ_{max} is very small in local patches when the surface colors are consistent. The maximum chromaticity σ_{max} in Figure 4.2 (b) is the same as the maximum diffuse chromaticity Λ_{max} except for specular pixels, which cause the intensity/color discontinuities within local patches of the same surface color. Intuitively, applying low-pass filtering to the maximum chromaticity σ_{max} in Figure 4.2 (b) will smooth out the variances due to specular highlights. However, there are two issues:

1. The smoothing filter should be edge-aware, such that the σ_{max} values of two pixels associated with different surface materials (Λ_{max} values are different) will not be blend together.

2. The diffuse pixels will be affected by the specular pixels after smoothing.

As a popular edge-aware operator, joint bilateral filter can be employed to smooth the maximum chromaticity σ_{max} using the maximum diffuse chromaticity Λ_{max} as the smoothing guidance. But Λ_{max} is to be estimated; thus one needs to find a substitution or an approximation. Although the "pseudo-coded" diffuse



(a)Input. (b)σ_{max}. (c)Pseudo diffuse.

(d)Diffuse ([31]). (e) $\sigma_{max} = \Lambda_{max}$. (f)$\lambda_{max}$.

(g)Diffuse(ours, 1 iter.). (h)Diffuse(ours, 3 iter.). (i)Diffuse(ours, 10 iter.).

Figure 4.2: Highlight removal on real data set.

image presented in [31] is free of specularity, it is not a good substitution for this problem because its color depends on both the surface geometry and material, while Λ_c is invariant to the surface geometry. Let

$$\sigma_{min} = \min(\sigma_r, \sigma_g, \sigma_b), \tag{4.10}$$

I approximiate Λ_c using λ_c computed as follows:

$$\lambda_c = \frac{\sigma_c - \sigma_{min}}{1 - 3\sigma_{min}}. \tag{4.11}$$

The relationship between the approximated diffuse chromaticity λ_c and the real diffuse chromaticity Λ_c is captured in Theorem 1 and Theorem 2.

Theorem 1 *For any two pixels $\vec{p}$ and $\vec{q}$, if $\Lambda_c(\vec{p}) = \Lambda_c(\vec{q})$, then $\lambda_c(\vec{p}) = \lambda_c(\vec{q})$.*

Proof: For any two pixels p and q, let

$$J_{min} = \min(J_r, J_g, J_b), \tag{4.12}$$
$$J^D_{min} = \min(J^D_r, J^D_g, J^D_b), \tag{4.13}$$

According to the definition of λ_c in Eqn. (4.11), I obtain

$$\begin{aligned}
\lambda_c &= (\sigma_c - \sigma_{min})/(1 - 3\sigma_{min}) \qquad (4.14)\\
&= (\frac{J_c}{\sum_{u\in\{r,g,b\}} J_u} - \frac{J_{min}}{\sum_{u\in\{r,g,b\}} J_u})\\
&/ \quad (\frac{\sum_{u\in\{r,g,b\}} J_u}{\sum_{u\in\{r,g,b\}} J_u} - \frac{3\cdot J_{min}}{\sum_{u\in\{r,g,b\}} J_u})\\
&= (\frac{J_c - J_{min}}{\sum_{u\in\{r,g,b\}} J_u})/(\frac{\sum_{u\in\{r,g,b\}}(J_u - J_{min})}{\sum_{u\in\{r,g,b\}} J_u})\\
&= \frac{J_c - J_{min}}{\sum_{u\in\{r,g,b\}}(J_u - J_{min})}\\
&= \frac{(J^D_c + J^S) - (J^D_{min} + J^S)}{\sum_{u\in\{r,g,b\}}((J^D_u + J^S) - (J^D_{min} + J^S))}\\
&= \frac{J^D_c - J^D_{min}}{\sum_{u\in\{r,g,b\}}(J^D_u - J^D_{min})}\\
&= (\frac{J^D_c - J^D_{min}}{\sum_{u\in\{r,g,b\}} J^D_u})/(\frac{\sum_{u\in\{r,g,b\}}(J^D_u - J^D_{min})}{\sum_{u\in\{r,g,b\}} J^D_u})\\
&= (\Lambda_c - \Lambda_{min})/(\sum_{u\in\{r,g,b\}}(\Lambda_u - \Lambda_{min})). \qquad (4.15)
\end{aligned}$$

For any two pixels p and q, if $\Lambda_c(p) = \Lambda_c(q)$, Eqn. (4.15) ensures that $\lambda_c(p) = \lambda_c(q)$.

Theorem 2 *For any two pixels $\vec{p}$ and $\vec{q}$, if $\lambda_c(\vec{p}) = \lambda_c(\vec{q})$, then $\Lambda_c(\vec{p}) = \Lambda_c(\vec{q})$ only if $\Lambda_{min}(\vec{p}) = \Lambda_{min}(\vec{q})$.*

Proof: For any two pixels p and q, assume $\lambda_c(p) = \lambda_c(q)$; from Equ. (4.15) I obtain

$$\frac{\Lambda_c(p) - \Lambda_{min}(p)}{1 - 3\Lambda_{min}(p)} = \frac{\Lambda_c(q) - \Lambda_{min}(q)}{1 - 3\Lambda_{min}(q)}. \tag{4.16}$$

If $\Lambda_{min}(p) = \Lambda_{min}(q)$, from (4.16) I obtain $\Lambda_c(p) = \Lambda_c(q)$. Note that λ_c is just an approximation of Λ_c, which will fail for the specific case specified in Theorem

2. However, it is the best estimate I can get. Figure 4.2 (f) presents the maximum values of the approximated diffuse chromaticity

$$\lambda_{max} = \max(\lambda_r, \lambda_g, \lambda_b) \tag{4.17}$$

$$= \max(\frac{\sigma_r - \sigma_{min}}{1 - 3\sigma_{min}}, \frac{\sigma_g - \sigma_{min}}{1 - 3\sigma_{min}}, \frac{\sigma_b - \sigma_{min}}{1 - 3\sigma_{min}}) \tag{4.18}$$

computed from the input image presented in Figure 4.2 (a).

Using the approximated maximum diffuse chromaticity defined in Eqn. (4.18) to guide the smoothing, the filtered maximum chromaticity σ_{max} can be computed as follows:

$$\sigma^F_{max}(\vec{p}) = \frac{\sum_{\vec{q}\in\Omega} \mathcal{F}(\vec{p},\vec{q})\mathcal{G}(\lambda_{max}(\vec{p}), \lambda_{max}(\vec{q}))\sigma_{max}(\vec{q})}{\sum_{\vec{q}\in\Omega} \mathcal{F}(\vec{p},\vec{q})\mathcal{G}(\lambda_{max}(\vec{p}), \lambda_{max}(\vec{q}))}, \tag{4.19}$$

where $\mathcal{F}$ and $\mathcal{G}$ are spatial and range weighting functions which are typically Gaussian in the literature [60], [61].

The variances of the maximum chromaticity σ_{max} due to specular highlights will be reduced after filtering, and the filtered maximum chromaticity σ^F_{max} will be closer to Λ_{max} than σ_{max} for the specular pixels. However, after smoothing, the diffuse pixels will be affected by the specular pixels too. According to Theorem 3, the filtered maximum chromaticity values σ^F_{max} of the diffuse pixels will be lower than the un-filtered values σ_{max}. As a result, to exclude the contribution of the specular pixels, I compare σ^F_{max} and σ_{max} and take the maximum value:

$$\sigma_{max}(\vec{p}) = \max(\sigma_{max}, \sigma^F_{max}(\vec{p})). \tag{4.20}$$

Theorem 3 *Assume* $\Gamma_c = \frac{1}{3}$*, then* $\Lambda_{max} \geq \sigma_{max}$*. Equality holds when the* $\Lambda_{max} = \frac{1}{3}$.

Proof: let $J_{max} = \max(J_r, J_g, J_b)$, according to the definition of σ_{max},

$$\begin{aligned}
\sigma_{max} &= \frac{J_{max}}{\sum_{u\in\{r,g,b\}} J_u}, && (4.21)\\
&= \frac{J^D_{max} + J^S}{\sum_{u\in\{r,g,b\}} J^D_u + 3J^S} && (4.22)\\
&= (\Lambda_{max} + \frac{J^S}{\sum_{u\in\{r,g,b\}} J^D_u})/(1 + \frac{3J^S}{\sum_{u\in\{r,g,b\}} J^D_u}) && (4.23)\\
&= \Lambda_{max} + \frac{J^S}{\sum_{u\in\{r,g,b\}} J^D_u} - \sigma_{max}\frac{3J^S}{\sum_{u\in\{r,g,b\}} J^D_u} && (4.24)\\
&= \Lambda_{max} + (1 - 3\sigma_{max})\frac{J^S}{\sum_{u\in\{r,g,b\}} J^D_u}. && (4.25)
\end{aligned}$$

According to its definition, $\sigma_{max} \geq 1$; thus,

$$\sigma_{max} - \Lambda_{max} \leq 0, \tag{4.26}$$

and the quality holds when the $\Lambda_{max} = \sigma_{max} = \frac{1}{3}$.

I then iteratively apply joint bilateral filter to σ_{max} such that the maximum diffuse chromaticity values can be gradually propagated from the diffuse pixels to the specular pixels. In practice, I compare the filtered values σ^F_{max} with σ_{max} after every iteration. The algorithm is believed to converge when their difference is smaller than a threshold (set to 0.03 in our experiments) at every pixel. Our method generally converges after $2-3$ iterations. Figure 4.2 (g)-(i) present the extracted diffuse reflections after 1, 3 and 10 iterations, respectively. The proposed highlight removal algorithm is summarized in Algorithm 1.

4.3.2 Iterative Depth and Diffuse Reflection Estimation

In this section, I represent an iterative optimization framework to estimate the depth values and the diffuse reflections in the first and second modules as shown in Figure 4.1. For each iteration $i = \{1, 2, \cdots, K\}$, multi-view stereo matching is first performed on the current diffuse reflection hypotheses $\vec{I}^{(i-1)}_d$ generated by the first module. A set of depth maps $\vec{D}^{(i)}$ is obtained after stereo matching. The depth maps are then fused to give a coherent 3D reconstruction using the method presented in [59, 43]. Note that in the first iteration, the saturated pixels (identified using confidence map $\vec{C}^{(0)}$) are excluded from processing. Based on the

Algorithm 1 Highlight removal using a single RGB image

1: Compute σ_{max} at every pixel using the input image and store it as a grayscale image.
2: Compute λ_{max} at every pixel using the input image and store it as a grayscale image.
3: **repeat**
4: -Apply joint bilateral filter to image σ_{max} using λ_{max} as the guidance image (Eqn. 4.20), store the filtered image as σ^F_{max};
5: -For each pixel $\vec{p}$,

$$\sigma_{max}(\vec{p}) = \max(\sigma_{max}(\vec{p}), \sigma^F_{max}(\vec{p})); \tag{4.27}$$

6: **until** $\sigma^F_{max} - \sigma_{max} < 0.03$ at every pixel.

fused depth maps $\vec{D}^{(i)}_f$, the current diffuse reflection hypotheses are fused to reject outliers due to the errors presented in the first module (reflection components separation from single view). Each time, one of the cameras is selected as the reference camera. The current diffuse reflection hypotheses ($\vec{I}^{(i-1)}_d$) are projected to the reference camera. For each pixel in the reference camera, an array of color vectors is collected from the projected colors. The median color vector is accepted as correct. Similar to depth map fusion, the saturated pixels are not processed in the first iteration. After diffuse reflection fusion, the current diffuse reflection hypotheses of the saturated pixels ($\vec{I}^{(i)}_d$) are fed back to the first module, and another iteration begins. During this iteration, I assume that there is no saturated pixel by changing the status of every pixel to be confident: $\vec{C}^{(i)} = 1, i >= 1$. Our experiments show that in most cases two iterations are enough for convergence.

4.4 Experiments

4.4.1 Synthetic Data Set

To quantitatively evaluate the proposed method, I created a synthetic multi-color teapot with a directional light source with illumination chromaticity $\{0.32, 0.37, 0.31\}$. I increase the difficulty of the problem to be solved by assuming that the response function of the camera is not exactly linear:

$$f(J) = (J)^{0.9}, \tag{4.28}$$

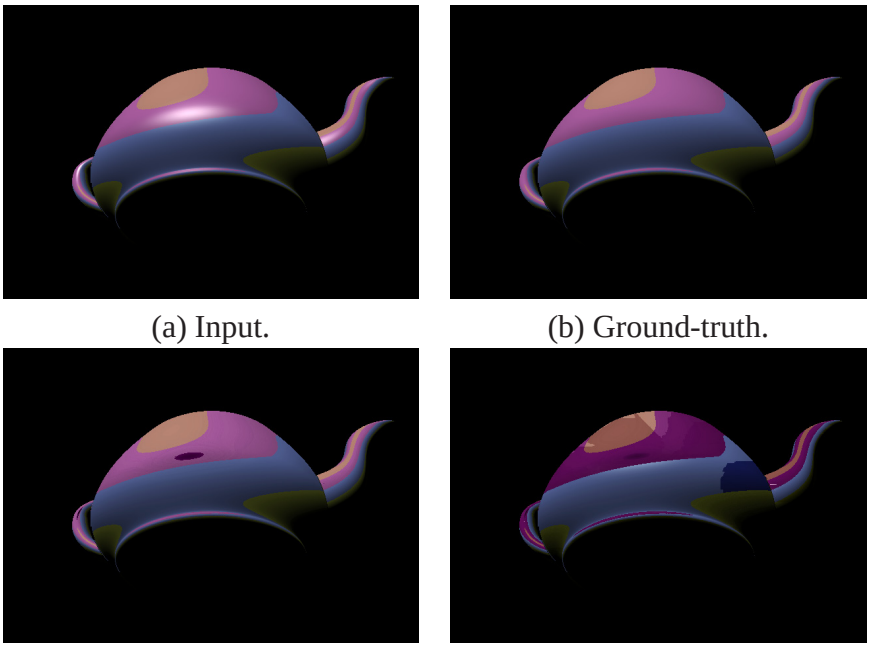
Figure 4.3: Comparison of our BFHR method and PHR method.

where J is the synthetic shading value using Phong shading model [52]. The nonlinearity violates the assumptions of all methods presented in Table 4.1 except [40], and may cause serious problems for them. Also, to approximate the real environment, the synthetic images generally contain large areas of saturated pixels. I show that our method is more robust in these situations. A total of 20 synthetic images were used in the experiment.

Table 4.1: Comparison with the most related methods.

Method	Assumptions					Output	
	No Saturation	Sparse Specularity	Single Material	Lighting Calibration	Object centered	Reflection Separation	Depth
[32, 30, 31]	X					X	
[49]	X		X		X	X	X
[50]	X		X		X	X	X
[51]	X		X	X	X	X	X
[40]		X				X	
Ours						X	X

I first compared our BFHR method with the PHR method [31] in Figure 4.3 using the peak signal-to-noise ratio (PSNR). For two intensity images $I_1, I_2 = [0; 1]$, this ratio is defined as $10 \log_{10}((h \cdot w)/\sum_p |I_1(p) - I_2(p)|^2)$, where h and w are the height and the width of image I_1 and I_2, and p is one of the pixels. PSNR values above 40 dB correspond to almost invisible differences [62]. Note that both BFHR and PHR require a single image and are invalid for saturated pixels. As the index of the image increases, the number of saturated pixels increases, and

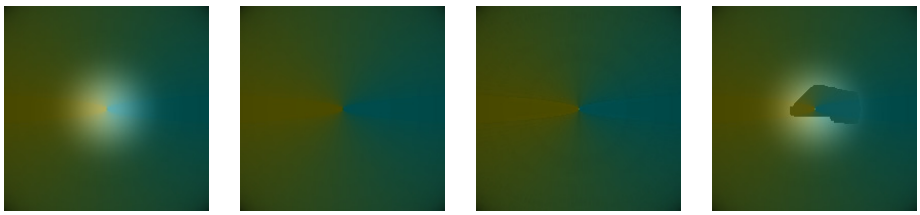


(a) Input. (b) Ground-truth.

(c) BFHR (PSNR=37.9dB). (d) PHR (PSNR=24.1dB).

Figure 4.4: Comparison of our BFHR method and PHR method.

the performance of both methods decreases. Visual comparison of the two methods is presented in Figure 4.4 and 4.5. The PHR method separates the diffuse and specular components by iteratively propagating the chromaticity information from diffuse pixels to specular pixels. It has the non-convergence problem as can be seen in Figure 4.4 (d) and 4.5 (d). Figures 4.4 (d) and 4.5 (d) also show that the PHR method extracts inaccurate diffuse reflections. Our BFHR method does not have these problems. The specular reflection separation error due to saturation

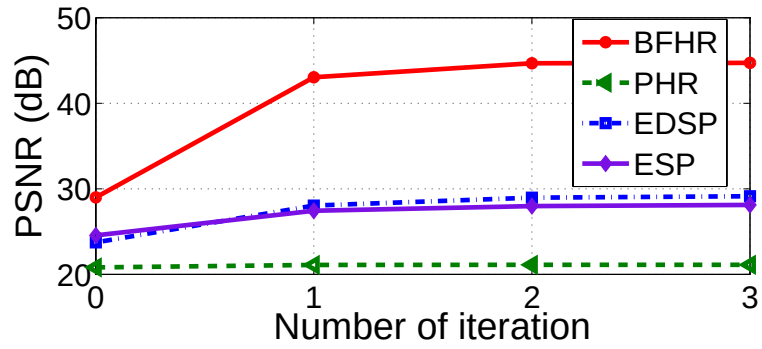


(a) (b) (c) (d)

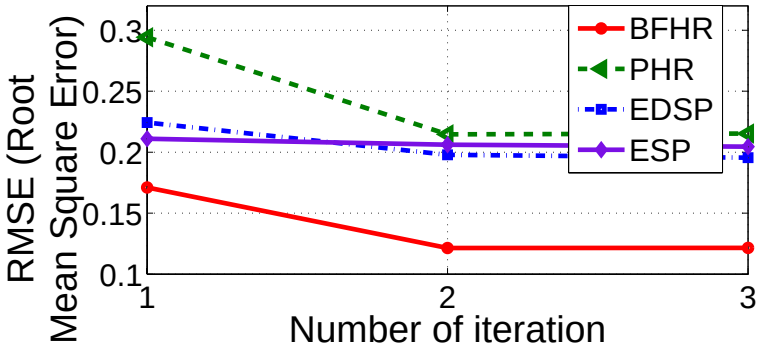
Figure 4.5: Comparison of our BFHR method and PHR method using a highly textured scene.

and nonlinearity can be removed using our iterative depth and diffuse reflection estimation framework. Figure 4.6 (a) shows how our method iteratively refines the

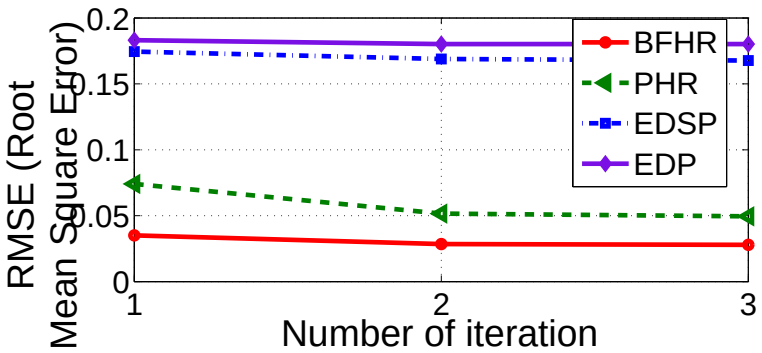
estimated diffuse reflection using PSNR, and that our method always converges in 2 iterations. Note that the PSNR values after 0 iteration correspond to the diffuse reflections generated by the first module (reflection components separation from single view). Any highlight removal method can be used in the first module. Figure 4.6 (a) quantitatively evaluates four such methods. The red curve corre-



(a) Highlight removal.



(b) Stereo matching.



(c) Depth map fusion.

Figure 4.6: Quantitative evaluation of the proposed iterative depth and diffuse reflection estimation approach.

sponds to our BFHR method, green curve to the PHR method [31], the blue curve

to the method in [40] (excluding the detected specular pixels (EDSP) from processing), and the purple curve to the simplest method with saturated pixels (ESP) excluded from processing. It is apparent that our iterative framework performs best with our BFHR method. Note that after only 1 iteration, the PSNR values are over 40 dB, which means that there are almost no visible differences with the ground truth.

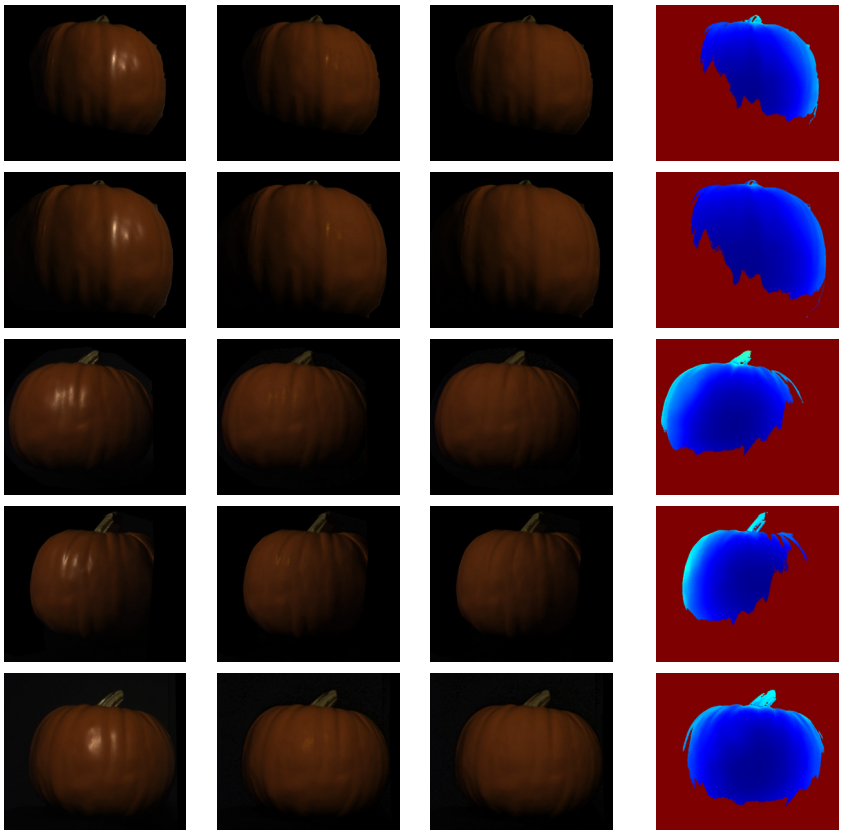
Our method simultaneously estimates the depth maps using multi-view stereo matching and fusion. The numerical evaluation is provided in Figure 4.6 (b) and (c). As can be seen, our iterative framework improves the performance of all methods, and our BFHR method achieves the highest performance. The error is mainly due to lack of texture and inaccurate diffuse color estimation.

4.4.2 Real Data Set

A total of 24 real images were captured with a Sony DFW-SW900 camera. A pumpkin-like object was placed on a table together with a calibration pattern. All the objects were put in a dark room illuminated by a spotlight to minimize ambient light. Distributed light introduces noise in highlight removal as the illumination chromaticity will not be accurate. Although our method is theoretically built on correct illumination chromaticity and linear camera response function assumptions, it is robust to violations of these assumptions.

The experimental results are presented in Figure 4.7. Row (a) shows 5 out of 24 input images, and (b) shows the extracted diffuse components using our BFHR method. There are noticeable errors due to saturation and nonlinear camera response function. Row (c) shows the refined diffuse images, and the estimated depth maps are presented in (d). As can be seen, the noticeable errors in (b) are eliminated in (c). Note that although this data set is of low texture, the shading information makes the stereo matching possible [47]. Additionally, depth map fusion [43] greatly improves the reconstruction quality.

I also tested our method with images provided by [31]. These images are captured by a SONY DXC-9000 (a progressive three CCD digital camera) by setting the gamma off, and the objects were lit with a solux halogen lamp. As a result, there is very little noise (e.g., inaccurate illumination chromaticity due to distributed light and saturation). Both our method and PHR can effectively remove the highlights as shown in Figures 4.8 and 4.9.



(a)Input. (b)BFHR. (c)Refined. (d)Fused Depth.

Figure 4.7: Experiments with real images.

4.5 Discussion

In this chapter, I describe a new highlight removal method based on bilateral filtering by assuming that the input images have chromatic surfaces and the illumination chromaticity is known or can be calibrated/estimated. I then present an iterative optimization approach for simultaneously estimating the depth and diffuse reflection via multi-view stereo matching and depth map fusion. I iteratively reduce reflection separation errors due to saturation, and depth estimation error due to incorrect reflectance estimates.

However, if the assumption is violated, our method generally fails. Such an example is provided in Figure 4.10. Neither our method nor the PHR method in [31] works for the white flowers in Figure 4.10, which are neutral. However, our method is more robust to the noise introduced by incorrect illumination chromaticity estimation. As a JPEG image captured with normal camera under multiple

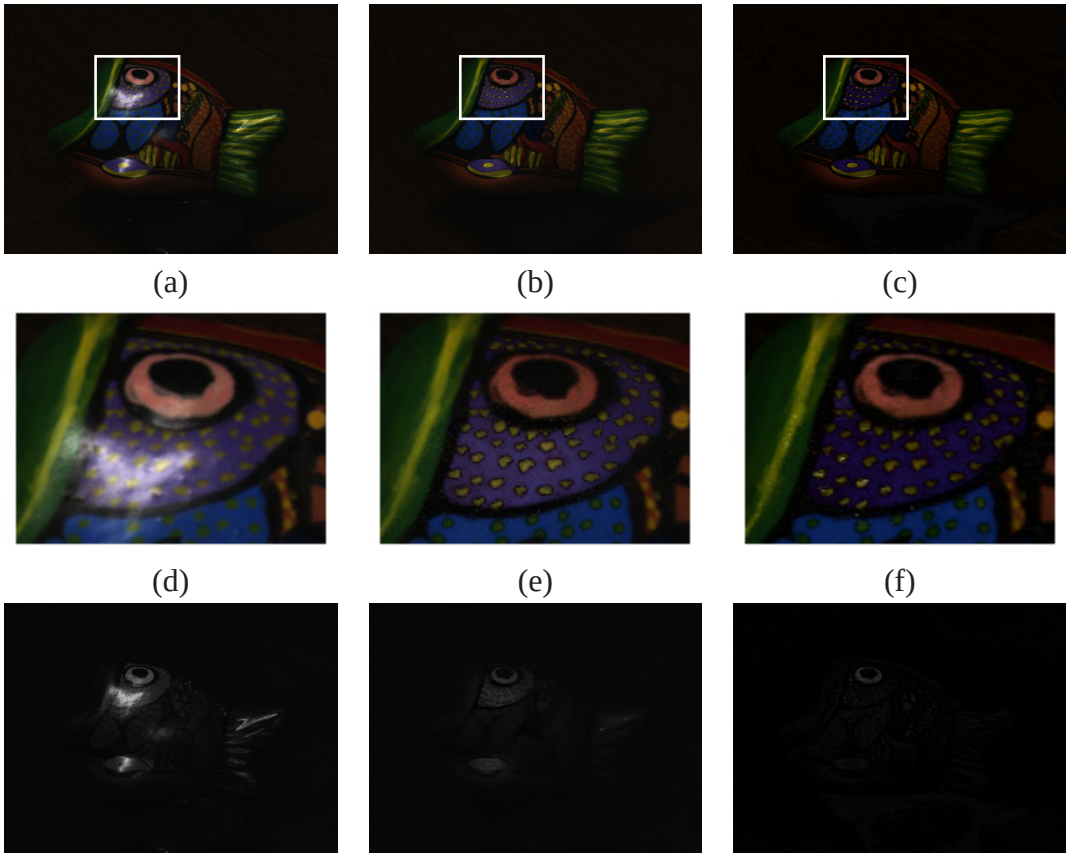


Figure 4.8: Experiments with real images.

flashes, Figure 4.10 (a) experiences quantization noise and the brightness values are not linearly related to the flux of incoming light. This noise causes the PHR method in [31] to fail. In Figure 4.10, (c) is almost the same as (d), which shows that the extracted diffuse image in (c) is incorrect. Because the maximum chromaticity of (d) is the same for every pixel, however, the maximum chromaticity of the diffuse component of (a) is not a constant.

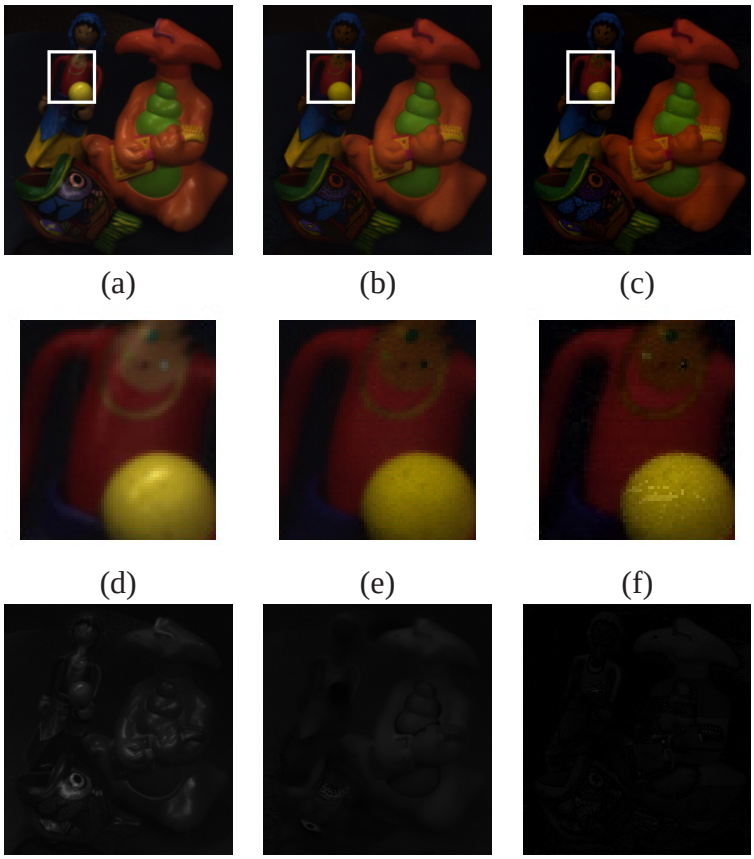


Figure 4.9: Experiments with real images.

Figure 4.10: Limitation of our method.

CHAPTER 5

REAL-TIME $O(1)$ BILATERAL FILTERING

5.1 Introduction

Originally introduced by Tomasi and Manduchi [60], bilateral filters are edge preserving operators that have found widespread use in many computer vision and graphics tasks like tone management [61, 68], denoising [23, 62, 63, 64, 65, 66], texture editing and relighting [67], demosaicking [69], stylization [70], optical-flow estimation [71, 72] and stereo matching [3, 12].

Until recently, bilateral filters were too computationally intensive for real-time applications. Several efficient numerical schemes [61, 62, 73, 74, 75] enable it to be computed at interactive speed or even video rate using GPU (graphics processing unit) implementation [76]. With the exception of [62], which approximates the bilateral by filtering subsampled copies of the image, these algorithms do not scale well since they become more expensive as the filtering window size grows, which limits their utility in high resolution real-time applications. The algorithm in [62] actually becomes faster as the size increases due to greater subsampling, but the exact output is dependent on the phase of subsampling.

It was therefore a significant advance when Porikli [77] demonstrated that bilateral filters can be computed at constant time with respect to filter size for three types of bilateral filters. (1) Box spatial and arbitrary range kernels. Integral histogram is used to avoid the redundant operations and interactive speed is achieved by quantizing the input image using a small number of bins, thus trading memory footprint and image quality for speed. For an 8-bit grayscale image, assuming 256 bins are used to compute and store the integral histogram, memory $256\times$ that of the image is required. The memory could be reduced, but that would also change single integral histogram computation to be 256 times, which is much slower. (2) Arbitrary spatial[1] and polynomial range kernels. A bilateral filter of this form can

[1] An IIR O (1) solution needs to be available for the kernel.

be interpreted as the weighted sum of the spatial filtered responses of the powers of the original image. No approximation is used in this method. (3) Arbitrary spatial and Gaussian range kernels. Taylor series is used to approximate the Gaussian range function up to the four order derivatives. However, this method is a bad approximation for small Gaussian variances.

A new $O(1)$ bilateral filtering method extending Durand and Dorsey's piecewise-linear bilateral filtering method [61] is proposed in the chapter. As in [61], I discretize the image intensities into a number of values, and compute a linear filter for each such value, the output of which is defined as Principle Bilateral Filtered Image Component (PBFIC) in this chapter. The final output is then a linear interpolation between the two closest PBFICs. Instead of confining the kernels to be Gaussian spatial and Gaussian range and using fast Fourier transform (FFT) for Gaussian convolution, which has cost $O(\log r)$ (r is the filter radius), I show that the discretization method can be directly extended to obtain $O(1)$ bilateral filtering with arbitrary spatial and arbitrary range kernels assuming that the exact or approximated spatial filter can be computed at constant time, e.g., box filtering using integral image [78], Gaussian filtering using recursive filtering [79], and Polynomial filtering using a set of integral images and more as presented in [77]. I also extend the method for $O(1)$ median filtering using integral image.

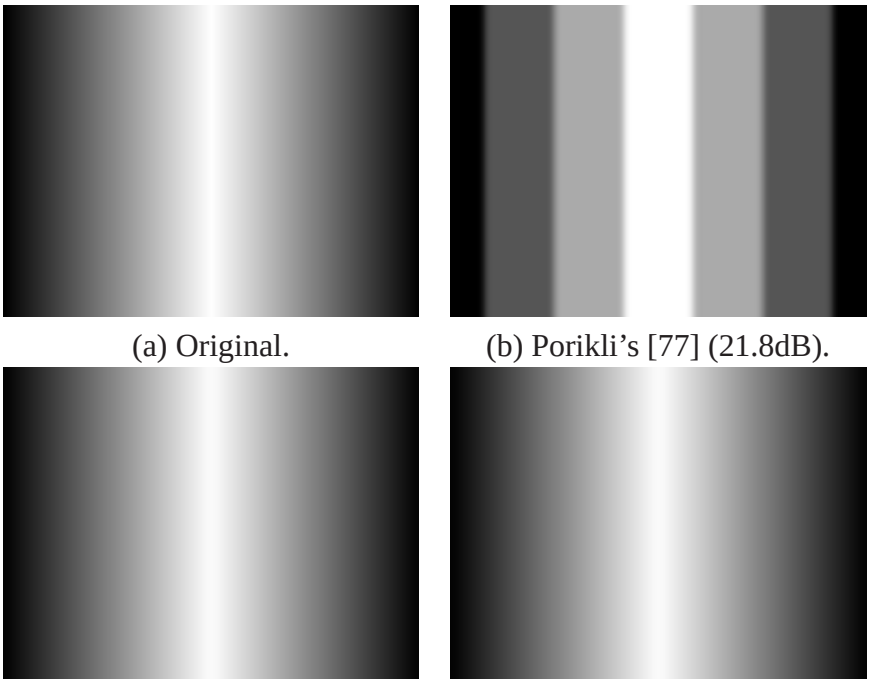
The contribution of this chapter is a new algorithm for constant time bilateral filtering with the following advantages over the state of the art [77]:

1. Uniform framework for constant time bilateral filtering with *arbitrary spatial and arbitrary range* kernels. In [77], only three types of $O(1)$ bilateral filtering are available.

2. Better Gaussian range function representation. Although the range function is quantized in our method, it is valid for both low and high variance Gaussian. The bilateral filtering method with arbitrary spatial and Gaussian range kernels presented in [77] uses Taylor series approximation, which is a bad approximation for low variance Gaussian. It is important to note that many applications require low range variance to preserve edges.

3. More accurate. Our method only quantizes the range function, while in [77], image intensities are also quantized, resulting in lower accuracy as shown in Figure 5.1.

4. Faster ($10\times$). Our method can be easily implemented in parallel. On the



(a) Original. (b) Porikli's [77] (21.8dB).

(c) Ours (66.2dB). (d) Exact.

Figure 5.1: Robustness to quantization.

NVIDIA Geforce 8800 GTX GPU, I show that given the same output accuracy, our method can be about $10\times$ faster on average.

5. Lower memory requirement (2%) enabling processing of high resolution images/videos. For box spatial bilateral filtering with 8-bit grayscale images, to obtain the exact bilateral filtering results, our method only requires about $4\times$ the memory of the image, while [77] requires $256\times$ the image memory for computing and storing the integral histogram (C implementation of our method is provided at the author's homepage [80]).

6. Extension of the O(1) framework for cross/joint bilateral filtering and median filtering.

The effectiveness of the proposed method is then experimentally verified for a variety of applications including natural video conferencing, interactive filtering, image/video abstraction, highlight removal, and multi-focus.

5.2 $O(1)$ Bilateral Filtering with Arbitrary Spatial and Arbitrary Range Kernels

A bilateral filter generally contains a spatial and a range filter kernel. Denote $\mathbf{x}$ as a pixel in the image and $\mathbf{y}$ as a pixel in the neighborhood $N(\mathbf{x})$ of $\mathbf{x}$: $I(\mathbf{x})$ and $I(\mathbf{y})$ are the corresponding range values of pixels $\mathbf{x}$ and $\mathbf{y}$. The filtered range value of $\mathbf{x}$ is

$$I^B(\mathbf{x}) = \frac{\sum_{\mathbf{y}\in N(\mathbf{x})}(f_S(\mathbf{x},\mathbf{y})\cdot f_R(I(\mathbf{x}),I(\mathbf{y}))\cdot I(\mathbf{y}))}{\sum_{\mathbf{y}\in N(\mathbf{x})}(f_S(\mathbf{x},\mathbf{y})\cdot f_R(I(\mathbf{x}),I(\mathbf{y})))}, \tag{5.1}$$

where f_S and f_R are the spatial and range filter kernels, respectively. If the range function is computed based on another image D where the range values of pixels $\mathbf{x}$ and $\mathbf{y}$ are $D(\mathbf{x})$ and $D(\mathbf{y})$, the spatial filtered range value of pixel $\mathbf{x}$ for image I is

$$I^B_D(\mathbf{x}) = \frac{\sum_{\mathbf{y}\in N(\mathbf{x})}(f_S(\mathbf{x},\mathbf{y})\cdot f_R(D(\mathbf{x}),D(\mathbf{y}))\cdot I(\mathbf{y}))}{\sum_{\mathbf{y}\in N(\mathbf{x})}(f_S(\mathbf{x},\mathbf{y})\cdot f_R(D(\mathbf{x}),D(\mathbf{y})))}, \tag{5.2}$$

and the resulting filter is called a cross (or joint) bilateral filter [68, 81], which enforces the texture of filtered image I^B_D to be similar to image D.

5.2.1 Decomposing a Bilateral Filter into Spatial Filters

I review Durand and Dorsey's piecewise-linear bilateral filtering method in this section and show that it can be directly extended for $O(1)$ bilateral filtering with arbitrary spatial and arbitrary range kernels.

In practice, the pixel intensity for an image $I(\mathbf{x})$ is discrete with $I(\mathbf{x}) \in \{0,\cdots,N-1\}$, where N is the total number of grayscale values. Letting $I(\mathbf{x}) = k$, Equation (5.1) can be expressed as

$$I^B(\mathbf{x}) = \frac{\sum_{\mathbf{y}\in N(\mathbf{x})}(f_S(\mathbf{x},\mathbf{y})\cdot f_R(k,I(\mathbf{y}))\cdot I(\mathbf{y}))}{\sum_{\mathbf{y}\in N(\mathbf{x})}(f_S(\mathbf{x},\mathbf{y})\cdot f_R(k,I(\mathbf{y})))}. \tag{5.3}$$

For every pixel $\mathbf{y}$ and every intensity value $k \in \{0,\cdots,N-1\}$, define

$$W_k(\mathbf{y}) = f_R(k, I(\mathbf{y})) \tag{5.4}$$

and

$$J_k(\mathbf{y}) = W_k(\mathbf{y})\cdot I(\mathbf{y}). \tag{5.5}$$

Bilateral filtering can then be decomposed into N sets of linear filter responses

$$J_k^B(\mathbf{x}) = \frac{\sum_{\mathbf{y}\in N(\mathbf{x})} f_S(\mathbf{x},\mathbf{y}) J_k(\mathbf{y})}{\sum_{\mathbf{y}\in N(\mathbf{x})} f_S(\mathbf{x},\mathbf{y}) W_k(\mathbf{y})} \tag{5.6}$$

so that

$$I^B(\mathbf{x}) = J^B_{I(\mathbf{x})}(\mathbf{x}), \tag{5.7}$$

where J_k^B is defined as Principle Bilateral Filtered Image Component (PBFIC) in this chapter. In practice, assume only $\hat{N}$ out of N PBFIC ($k \in \{L_0, \cdots, L_{\hat{N}-1}\}$) are used, and the intensity of pixel $\mathbf{x}$ is $I(\mathbf{x}) \in [L_k, L_{k+1}]$, the bilateral filtering value $I^B(\mathbf{x})$ can then be linearly interpolated from $J_k^B(\mathbf{x})$ and $J_{k+1}^B(\mathbf{x})$ as follows:

$$I^B(\mathbf{x}) = (L_{k+1} - I(\mathbf{x}))J_k^B(\mathbf{x}) + (I(\mathbf{x}) - L_k)J_{k+1}^B(\mathbf{x}). \tag{5.8}$$

Note that the range filter f_R is not constrained and any desired filter function can be chosen, but approximation can be poor if $\hat{N}$ is extremely small for some range filters, e.g., Gaussian filter.

According to Equation (5.4) and (5.5), the noise due to quantization only affects the range function W_k, and the pixel intensity values of the input image ($I(\mathbf{y})$ in Equation 5.5) will be preserved. However, both are quantized in the $O(1)$ bilateral filtering method presented in [77], and thus are less precise. The main computation of the method is $\hat{N} \times 2$ spatial filtering processing according to Eqn. (5.6). Additionally, in our method, image pixels are processed independently, allowing for parallel implementation. These are the two main reasons why our method outperforms the current state of the art [77] for both accuracy and speed. The main storage required is three memory buffers with the same size as the input image for images J_k^B, J_{k+1}^B and W_k. Note that J_k^B and J_k share the same memory buffer. Box/Gaussian spatial filtering also requires an additional memory buffer with the same size as the input image. Hence, the total memory buffer is about $4\times$ the size of the image memory. However, [77] requires a set of $\hat{N}$ image buffers to store the integral histogram during aggregation. Otherwise, the program will compute the integral histogram $\hat{N}$ times at the cost of speed.

5.2.2 $O(1)$ Spatial Filtering

As shown in Equation (5.6), bilateral filter with arbitrary spatial and range kernels can be decomposed into two sets of spatial filters on $J_k(\mathbf{y})$ and $W_k(\mathbf{y})$, respectively. The computation complexity thus depends on the complexity of spatial filtering. Enabling constant time spatial filtering results in constant time bilateral filtering with arbitrary range functions.

One of the most popular spatial filters is box filter, which can be easily computed in constant time using integral image [78] or summed area table [82]. Another popular spatial filter is Gaussian filter which if implemented in the Fourier domain is constant in the filter size. However, the discrete FFT and its inverse have cost $O(\log r)$, where r is spatial filter size. Hence, to achieve higher speed, I used Deriche's recursive method [79] to approximate Gaussian filtering which is able to run in constant time and the results are visually very similar to the exact. Polynomial filtering can also be computed in constant time $O(1)$ using a set of integral images [77]. More $O(1)$ spatial filters are presented in [77].

5.2.3 $O(1)$ Cross/Joint Bilateral Filtering and Median Filtering

In this section, I show that the decomposition method used for bilateral filtering can be easily extended for cross/joint bilateral filtering and median filtering. Changing $I(\mathbf{y})$ in Equation (5.4) to $D(\mathbf{y})$ and changing $I(\mathbf{x})$ in Equation (5.7) and (5.8) to $D(\mathbf{x})$ enables $O(1)$ cross/joint bilateral filtering, which enforces the texture of filtered image to be similar to image D.

For median filtering, assume

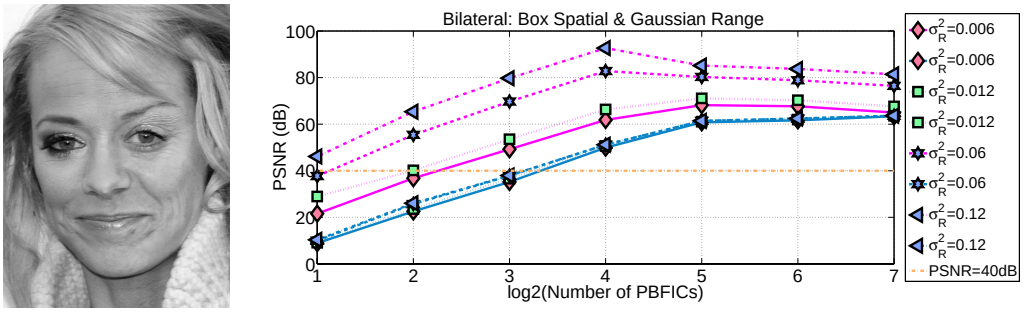
$$sign(x) = \begin{cases} -1 & x < 0 \\ 0 & x = 0 \\ 1 & x > 0 \end{cases}$$

and the intensity value is also in $[0, \ldots, N-1]$; the median filtered value $I^M(\mathbf{x})$ of a pixel $\mathbf{x}$ can then be expressed as

$$M_k(\mathbf{x}) = |\sum_{\mathbf{y}\in N(\mathbf{x})} sign(k - I(\mathbf{y}))| = |\sum_{\mathbf{y}\in N(\mathbf{x})} Q_k(\mathbf{y})|, \tag{5.9}$$

$$I^M(\mathbf{x}) = \underset{k\in\{0,1,\ldots,N-1\}}{\operatorname{argmin}} M_k(\mathbf{x}), \tag{5.10}$$

which shows that median filtering can be separated into two steps: (1) Box filtering is applied to a set of N images Q_k computed based on current intensity possibility k and the original image I. The absolute value of the box filtering results is computed as images M_k. (2) For each pixel, the intensity hypothesis $k \in \{0, 1, \ldots, N-1\}$ corresponding to the minimum pixel values of the images M_k is selected as correct. The box filtering in the first step depends on the filter size, but as shown in Section 5.2.2, it can be computed in constant time.



(a) Original image. (b) PSNR accuracy.

Figure 5.2: Effect of quantization on image quality.

5.3 Experimental Results

This section reports experimental results supporting our claims that our method advanced the state of the art. Like [77], I use the peak signal-to-noise ratio (PSNR) to evaluate numerical accuracy. For two intensity images I, $J \in [0, 1]$, this ratio is defined as $10 \log_{10}((h \cdot w)/\sum_{\mathbf{x}} |I(\mathbf{x}) - J(\mathbf{x})|^2)$, where h and w are the height and width of image I and J, and $\mathbf{x}$ is one of the pixels. It is assumed [62] the PSNR values above 40 dB often correspond to almost invisible differences.

Using Figure 5.2(a) as input image, the PSNR values for bilateral filtering with box spatial and Gaussian range ($f_R(I(\mathbf{x}), I(\mathbf{y})) = exp(-\frac{(I(\mathbf{x})-I(\mathbf{y}))^2}{2\sigma_R^2})/\sqrt{2\pi\sigma_R^2}$) kernels are presented in Figure 5.2(b) with respect to the number of PBFICs (bins) used. Figure 5.2 shows that our method is more accurate than Porikli's method [77] using both small and large range kernel variances. Also note that for our method, larger range variance results in much higher accuracy with fewer PBFICs, as the range function (Equation (5.4)) is more flat, and quantization introduces less noise. However, Porikli's method also quantizes the original image.

(a)Porikli's method. (b)Our method. (c)Exact.

(d)Porikli's method. (e)Our method. (f)Exact.

Figure 5.3: Performance under extreme quantization.

The improvement is thus much smaller. The filtered images with $\sigma_R^2 = 0.012$ and $\sigma_R^2 = 0.12$ using Porikli's method and our method are provided in Figure 5.3. As can be seen, our results are visually very similar to the exact even using very small number of PBFICs. To achieve acceptable PSNR value (> 40 dB) for variance $\sigma_R^2 \in [0.006, 0.12]$, our method generally requires 2 to 8 PBFICs, and the running time is about 3.7 ms to 15 ms for a 1 MB image. To achieve acceptable quality, Porikli's method requires a 16-bin integral histogram, and the total running time for constructing the integral histogram and computing the response for any given spatial filter size is about 75 ms; thus, our method is about $10\times$ faster on average. Also, our method is less memory-consuming. Only twice the memory of the original image is required by our method regardless of the number of PBFICs used. However, to obtain the same quality as exact, Porikli's method computes integral histogram using a total of 256 bins for an 8-bit grayscale image. Hence, $256\times$ the image memory is required. The heavy memory consumption can be avoided by repeatedly computing the integral histogram for every possible intensity value but

at the cost of speed.

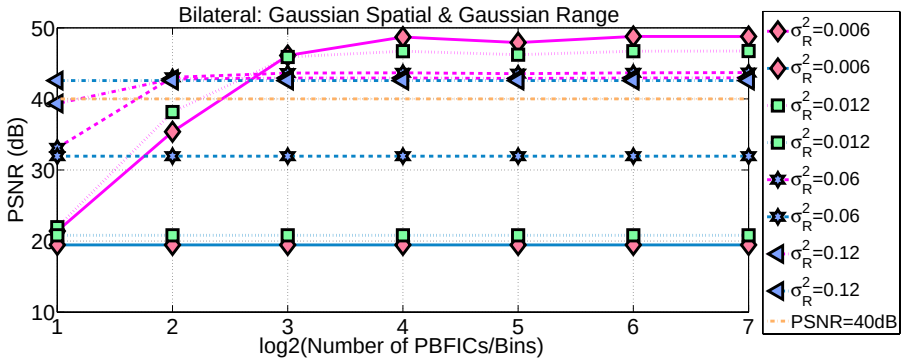
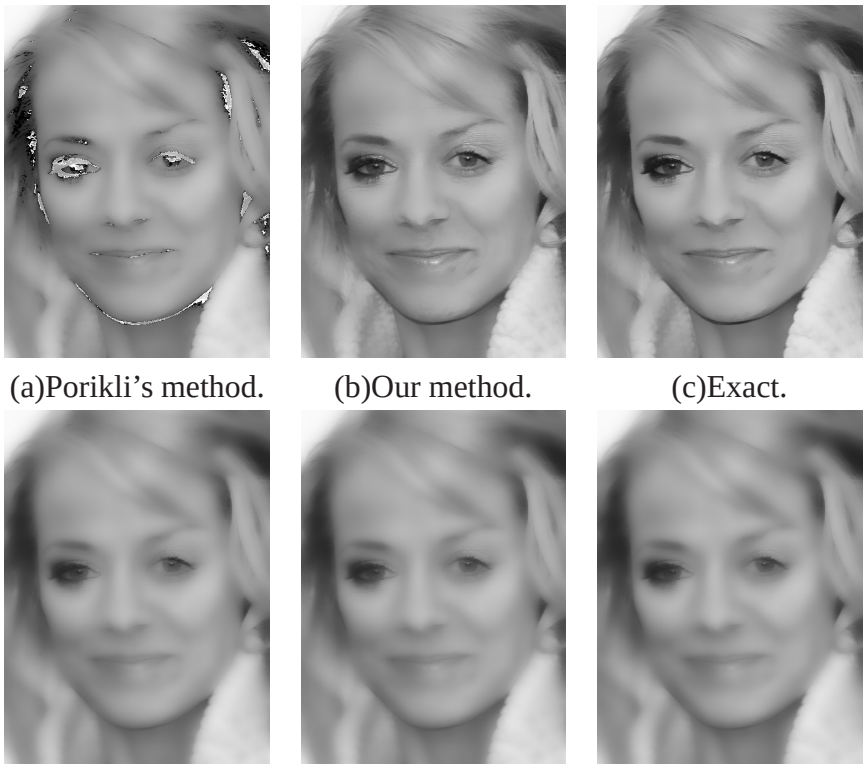


Figure 5.4: PSNR accuracy with respect to the number of PBFICs (bins) used.

Figure 5.4 presents the PSNR values of bilateral filtering with Gaussian spatial ($f_S(\mathbf{x},\mathbf{y}) = exp(-\frac{||\mathbf{x},\mathbf{y}||^2}{2\sigma_S^2})/\sqrt{2\pi\sigma_S^2}$) and Gaussian range kernels for the original image presented in Figure 5.3(a). Range variances ranging from 0.006 to 0.12 are tested. Obviously, Porikli's method (blue lines) fails for small range variances due to Gaussian approximation using Taylor series expansion. However, our method (pink curves) is valid for both small and large range variances. The corresponding filtered images are provided in Figure 5.5. For a typical 1 MB image, Porikli's method runs at about 0.35 seconds. Our GPU implementation runs at about 30 frames per second using 8 PBFICs (computational complexity of recursive Gaussian filtering is about twice that of the box filtering), which is above the acceptable threshold (> 40 dB) as shown in Figure 5.4. Hence, our method is about $10\times$ faster than Porikli's method. The memory requirement is similar for both methods since neither depends on the number of PBFICs (bins) used. Finally, experimental results on cross/joint bilateral filtering and median filtering are presented in Figure 5.6 and Figure 5.7.

5.4 Applications

In this section I demonstrate the usefulness of the constant time bilateral filtering operation for a variety of applications.



(a)Porikli's method. (b)Our method. (c)Exact.

(d)Porikli's method. (e)Our method. (f)Exact.

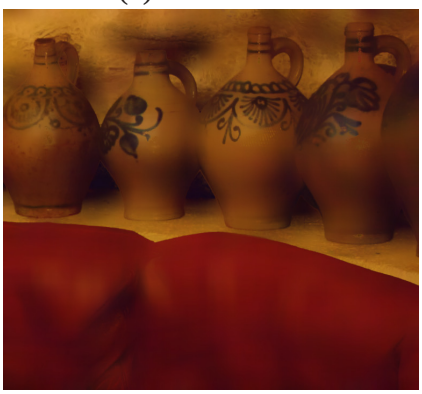
Figure 5.5: Bilateral filtering with Gaussian spatial and Gaussian range kernels.

5.4.1 Natural Video Conferencing

In modern video conference systems, e.g., Halo [83], the existing product line uses very high quality SD cameras that deliver natural, life-like images of meeting participants. With HD cameras and displays, even as they deliver additional sharpness, many unwanted details like wrinkles may also be amplified, resulting in images that may not be as pleasing as the SD images in today's product line. The constant time bilateral filtering method proposed provides a way to retain the salient features in HD images while removing unwanted details and noise. Figure 5.8 shows the result of applying our $O(1)$ bilateral filter to a portrait. The filtered result in Figure 5.8 (b) clearly preserves salient features while removing wrinkles. Linearly blending the original image with the filtered result produces Figure 5.8 (c), which is natural and realistic. The amount of blending can also be controlled in real time to deliver the most desirable output.

(a) Non-Flash.

(b) Flash.



(c) Filtered (47.9 dB).

(d) Filtered (Exact).

Figure 5.6: Joint bilateral filtering with box spatial (filter size: 51×51) and Gaussian range ($\sigma_R^2 = 0.006$) kernels.

(a) Original image

(b) Median filtering

Figure 5.7: $O(1)$ median filtering. (a) is the original image and (b) is our median filtering result which is the same as exact.

(a) Original. (b) Filtered. (c) Blend.

Figure 5.8: Natural video conferencing.

(a) Intermediate. (b) Intermediate. (c) Final.

Figure 5.9: Local bilateral filtering.

5.4.2 Interactive Filtering

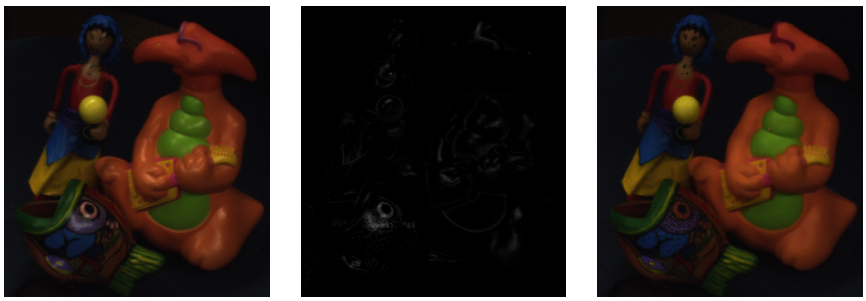
As shown in Figure 5.8 (b), full automatic/global bilateral filtering removes unwanted details, e.g., wrinkles. Unfortunately, some interesting details, e.g., hair, are lost. A human-guided local bilateral filtering method is then developed. The user is provided with a virtual brush. Filtering is applied locally to the regions where the brush passes by. The edge-preserving property guarantees a natural look after local filtering and the real-time speed enables human-compute interaction. Intermediate results are presented in Figure 5.9 (a) and (b), and the final result in Figure 5.9 (c).

5.4.3 Other Applications

Bilateral filtering and cross/joint bilateral filtering can also be used for image/video abstraction [70], highlight removal and multi-focus. Experimental results using our $O(1)$ bilateral filtering method are presented in Figure 5.10, 5.11 and 5.12, respectively.

(a) Original (512×384). (b) Abstracted.

Figure 5.10: Image/video abstraction.



(a) Original. (b) Specular. (c) Diffuse.

Figure 5.11: Highlight removal.

(a) Focus 1. (b) Focus 2.

(c) Focus 2. (d) Multi-Focus.

Figure 5.12: Multi-focus.

5.5 Discussion

A uniform framework enabling real-time $O(1)$ bilateral filtering with *arbitrary spatial and arbitrary range* kernels and parallel implementation is presented in the chapter. Experimental results show that our method outperforms the state of the art [77] for accuracy, speed and memory consumption. For bilateral filter with arbitrary spatial and Gaussian range kernels, our method works for both small and large range variances, while the method presented in [77] is invalid for small variances due to Taylor series approximation. Our framework can be easily extended for $O(1)$ cross/joint bilateral filtering and median filtering. Experimental results on a variety of applications verify the effectiveness of the proposed method.

In the future, I plan to implement the proposed method with NVIDIA Geforce $9800GX2$ GPU, which has twice the texture fill rate as the $8800GTX$ GPU used

in the chapter, and has the potential to double the speed reported. Meanwhile, [61] demonstrated that it is safe to use a downsampled version of the image except for the final interpolation (Eqn. 5.8). There is not any visible artifact up to downsampling factor of 10 to 25. I plan to implement this method which has the potential to improve the speed more than $10\times$.

CHAPTER 6

SVM FOR EDGE-PRESERVING FILTERING

6.1 Introduction

The objective of bilateral filtering is to smooth images. It is done by replacing the intensity (color) value of a pixel by the average of the values of other pixels weighted by their spatial distance and intensity similarity to the original pixel. Zucker et al. [84] used this idea by identifying similar pixels by first detecting edges. They iteratively replaced the intensity of a pixel by the average of all the pixels in a small (3×3) neighborhood, and on the same side of the edge as the pixel itself. Davis and Rosenfeld [85] identified similar pixels differently by choosing, from the nine pixels in the neighborhood, those six that are closest in intensity to the original pixel, and used their median to obtain the smoothed value. Ahuja [18] proposed a transform to compute the net similarity between a pixel and all other pixels in the image, as well as the direction in which the largest number of a pixel's most similar pixels are located. The latter was captured by computing a force vector at the pixel. Tabb and Ahuja [86] presented a detailed algorithm for multiscale image segmentation using the force transform and demonstrated the performance advantages of the similarity measure incorporated in the force transform. Tomasi and Manduchi [60] used the same definition of similarity as proposed by Ahuja [18] and used it for image smoothing. They replaced pixel values with similarity-weighted averages and called it bilateral filtering. Other than image smoothing and segmentation, bilateral filtering has found many other applications including denoising [23, 63, 64, 65, 66], texture editing and relighting [67], tone management [61, 68], demosaicking [69], stylization [70], optical-flow estimation [71, 72], and stereo matching [3, 12, 87, 88].

Bilateral filter is known to be computationally intensive. Recently, several methods [61, 73, 74, 75] have enabled it to be computed at either $O(r)$ or $O(log(r))$ runtime in the radius of the filter r. By filtering on the down-sampled image, Paris

and Durand [62] prove that the runtime of Durand and Dorsey's method [61] decreases as the filter size increases because the down-sampling factor can be increased without significantly impacting the accuracy of the result. This method is relatively slow when the filter size is small. Chen et al. [76] later show that the GPU implementation of [61] can achieve video rate.

Two $O(1)$ bilateral filtering methods have been proposed recently. Porikli [77] presents three types of $O(1)$ bilateral filters. (1) Box spatial and arbitrary range kernels. The problem with this method is that the spatial variances are not taken into account. Box-like artifacts may appear due to the imperfect frequency response of the spatial box filter. (2) Arbitrary spatial and polynomial range kernels. This method has not yet been demonstrated to be useful in practice due to poor edge preservation. (3) Arbitrary spatial and Gaussian range kernels. Taylor series is used to approximate the Gaussian range function up to the fourth order derivatives. However, this method is invalid for small Gaussian variances due to the limited approximation. In this chapter, I only compare the third method proposed in [77] since Gaussian range function is typically used in the literature. Yang et al. [56] show that using recursive Gaussian [79], Durand and Dorsey's method [61] takes constant time by decomposing bilateral filter into a number of constant time spatial filters (defined as Principle Bilateral Filtered Image Component (PBFIC) in the chapter). An $O(1)$ median filtering method is proposed based on the $O(1)$ bilateral filter in the chapter. The computational complexity of the bilateral filter is independent of the filter size but depends on the number of PBFICs used. The smaller the range variance value, the more PBFICs required, and the slower the method.

In this chapter, I model the bilateral filtering problem as a vector-mapping approximation and solve it using SVM regression. To our knowledge, this is the first learning-based bilateral filtering method. There are two steps involved in our method: training and predicting. The training step is processed off-line, in which an original image and the corresponding bilateral filtered image are used as input. I first apply Gaussian smoothing to the exponentiation (including exponent=0) of the original image. For each pixel, a corresponding feature vector is then constructed using the exponentiation of the image, the Gaussian filtered responses, and their products. The target value of the feature vector is the corresponding pixel value in the bilateral filtered image. The prediction step is processed on-line using the model produced in the training step. Each pixel is processed independently, which enables parallel implementation. The computational complexity of

the prediction step is invariant to the filter kernel size; i.e., it is $O(1)$ or constant time.

Compared with the other two $O(1)$ bilateral filtering methods [77, 56] , our method has the following advantages:

1. Most of the bilateral filter based applications require edge-preserving filtering in which low range variance Gaussian is indispensable. However, [77] is invalid for low range variance, and the runtime of [56] increases as the range variance decreases. Our method is valid for both high and low range variances and the runtime is independent of the range variance value.

2. Traditional bilateral filtering methods use the same range variance value across the image while our method can smooth the image in constant time with varying range variance values for every pixel. Based on this property, I propose a new bilateral filtering method (Section 6.2.1) avoiding the over-smoothing or under-smoothing artifacts in traditional bilateral filters.

3. To our knowledge, our method is the most efficient $O(1)$ bilateral filtering method yet developed. For a 1 MB grayscale image, the speed of our GPU implementation is about 473 frames per second on an Nvidia Geforce 8800GTX GPU. The computational complexity of our method is about half of [77]. The method in [56] exhibits quantization artifacts for low range variance Gaussian with the same computation complexity.

Additionally, I quantitatively evaluate our method and the other two $O(1)$ methods presented in [56] and [77] using 3638 images from six categories [89] (Section 6.3) and 485 images from two video sequences [90] (Section 6.4). This is different from the evaluation in [56] and [77], in which only a couple of images are tested. I believe that such a careful comparison should be clarified to the community since bilateral filtering has so many applications.

6.2 Approach

In this section, I present the details of our learning-based bilateral filtering method. The bilateral filter is a normalized convolution in which the weighting for each pixel q is determined by the spatial distance from the center pixel p, as well as

its relative difference in intensity. Let I_p be the intensity at pixel p and J_p be the filtered value,

$$J_p = \sum_{q \in \Omega} \mathcal{F}(p,q)\mathcal{G}(I_p, I_q)I_q / \sum_{q \in \Omega} \mathcal{F}(p,q)\mathcal{G}(I_p, I_q). \tag{6.1}$$

The spatial and range weighting functions $\mathcal{F}$ and $\mathcal{G}$ are often Gaussian in the literature [18, 60, 61]

$$\mathcal{F}(p,q) = exp(-(p-q)^2/(2\sigma_S^2)), \tag{6.2}$$

$$\mathcal{G}(I_p, I_q) = exp(-(I_p - I_q)^2/(2\sigma_R^2)), \tag{6.3}$$

where σ_S and σ_R are the spatial and range variances, respectively. In this chapter, I also confine the two weighting functions to be Gaussian.

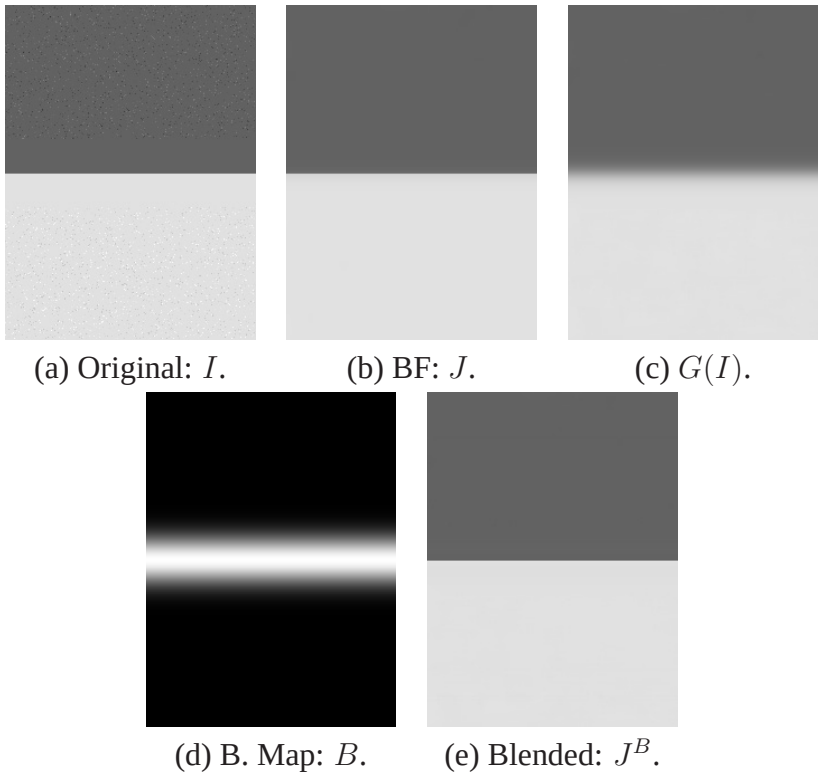
Figure 6.1 (b) shows the bilateral filtering result J of the original image I in (a), while (c) is obtained by setting the range weighting function $\mathcal{G}$ to be a constant, which is the Gaussian spatial filtered response $G(I)$ of the original image I in (a). Image (e) is obtained by combining (a) and (c) using the blending map B in (d); that is,

$$J_p^B = B_p \cdot I_p + (1 - B_p) \cdot G(I)_p, \tag{6.4}$$

where p is a pixel in the image. As can be seen, the noise in (a) is greatly reduced in both (b) and (e) while the intensity edges are well-preserved. The PSNR [62] value computed from (b) and (e) is 45 dB, which demonstrates that (b) and (e) are numerically similar to each other (PNSR$>$ 40 dB often corresponds to almost invisible differences as suggested in [62]). This experiment shows that image blending with a carefully designed blending map can be used to approximate bilateral filtering for simple scenes, and the blending process can be treated as a vector mapping process which maps a 2-dimentional vector $[I_p, G(I)_p]^T$ to a target value J_p^B.

However, for more complex scenes, for instance, the portrait in Figure 6.2 (a), it is hard to find a good blending map. Additionally, a simple combination between the original image (a) and the corresponding Gaussian filtered image (b) may not fit the bilateral filtered response correctly.

In this chapter, I also use the Gaussian filtered responses of the exponentiation (powers of pixel-wise intensities in the image) of the original image as shown



(a) Original: I. (b) BF: J. (c) $G(I)$.

(d) B. Map: B. (e) Blended: J^B.

Figure 6.1: Bilateral filtering and image blending.

from Figure 6.2 (b) to (e).[1] Similar to the image blending problem which can be cast as a 2D vector-mapping problem, I formulate the new problem as an N-dimensional vector-mapping problem. That is, from each pixel, an N-dimensional vector comprised of the values of exponentiation of the original image, the corresponding Gaussian filtered responses (as shown in Figure 6.2 (b) to (e)), and their products is obtained. Assuming that the exponent is up to n, $N = (n+1)^2 - 1$. For instance, assuming $n = 2$, the feature vector of pixel p is an 8-dimensional vector: $\mathbf{x}_p = [I_p, I_p^2, G(I)_p, G(I^2)_p, I_p G(I)_p, I_p^2 G(I)_p, I_p G(I^2)_p, I_p^2 G(I^2)_p]^T$. The N-dimensional feature vector is then mapped to the bilateral filtered value. In this chapter, the $\mathbb{R}^N$ to $\mathbb{R}^1$ mapping is denoted $\mathcal{M}$, that is: $\mathbb{R}^N \xrightarrow{\mathcal{M}} \mathbb{R}^1$. In practice, the mapping function $\mathcal{M}$ is unknown. In this chapter, I learn a robust mapping function from ϵ-Support Vector Regression [91]. Given a set of data points, $\{(\mathbf{x}_1, z_1); \cdots; (\mathbf{x}_l, z_l)\}$, such that $\mathbf{x}_p \in \mathbb{R}^N$ is an input and $z_p \in \mathbb{R}^1$ is a target output, the standard form of support vector regression [91] is used to solve the

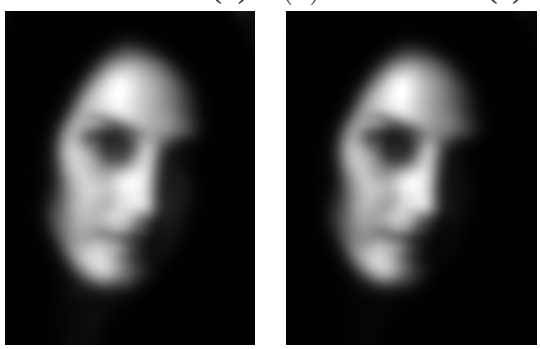
[1]The use of the powers of the image intensity is based on Taylor expansion approximation presented in [77].

following optimization problem:

$$\begin{array}{ll} \min_{\mathbf{w},b,\xi,\xi^*} & 0.5\mathbf{w}^T\mathbf{w} + C\sum_{p=1}^{l}(\xi_p + \xi_p^*) \\ \text{subject to} & \mathrm{w}^T\phi(\mathbf{x}_p) + b - z_p \leq \epsilon + \xi_p \\ & z_p - \mathrm{w}^T\phi(\mathbf{x}_p) - b \leq \epsilon + \xi_p^* \\ & \xi_p, \xi_p^* \geq 0, p = 1, \cdots, l. \end{array}$$

where ξ_p and ξ_p^* are slack variables that control the upper error bound, and C is a constant penalty factor to penalize data points $(\mathbf{x}_p, z_p)$ that do not satisfy $\mathrm{w}^T\phi(\mathbf{x}_p) + b - z_p \leq \epsilon$. Predictions within the error bound ϵ of the true target are not penalized. For all our experiments, $C = 10$ and $\epsilon = 0.05$, and the SVM-library [92] implementation with linear basis function kernels are used for the purpose of speed.

(a) Original: I. (b) $G(I)$. (c) $G(I^2)$.

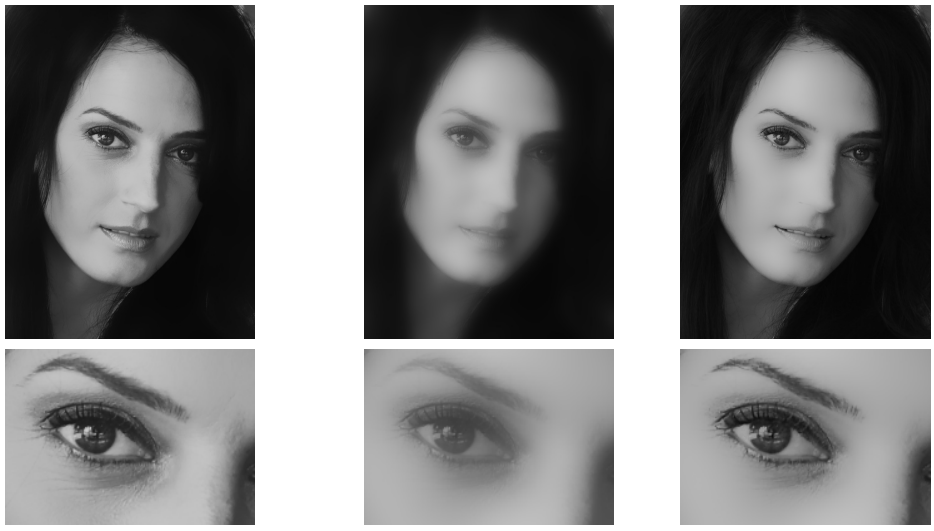


(d) $G(I^3)$. (e) $G(I^4)$.

Figure 6.2: Visual comparison of Gaussian filtered responses of the exponentiation of the original image.

6.2.1 $O(1)$ Bilateral Filter with Non-Uniform Range Variance

Traditional bilateral filtering methods depend on two fixed parameters (spatial and range variances: σ_S and σ_R) that indicate the size of the filter and the contrast of the features to preserve. However, fixing range variance σ_R results in either under-smoothing or over-smoothing artifacts as shown in Figure 6.3 (a) and (b), respectively. As can be seen, the unwanted wrinkles are not removed in (a) while the details around lip, eyes, and hair are lost in (b). Unlike previous methods, I fix the spatial variance σ_S but use varying range variance value σ_R for each pixel. I normalize the Gaussian filtered response of original image (Figure 6.2 (b)), such that the maximum value is a pre-defined value, and then use it as the range variance map. Note that our method is capable of filtering the image in constant time with non-uniform range variance values across the image by training different SVMs for each desired σ_R; thus it can be directly integrated with range variance map for non-uniform edge-preserving smoothing. The filtered image using our method with range variance $\sigma_R \in [0.01, 0.30]$ is presented in Figure 6.3 (c). That is, a total of 30 SVR functions corresponding to 30 σ_R values were trained, and at each pixel location, one of these SVR functions will be selected and used. It is apparent that, using our method, the wrinkles around the eyes are removed and the details of the lip, eyes, and hair are preserved.



(a) Exact BF ($\sigma_R = 0.05$). (b) Exact BF ($\sigma_R = 0.3$). (c) Our method.

Figure 6.3: $O(1)$ bilateral filter with non-uniform range variance.

6.2.2 Computational Complexity

The main computation involved in our method is the Gaussian filtering of the exponentiation of the original image. Figure 6.4 presents the numerical analysis of the behavior of our method with exponents from 2 to 5 w.r.t. the range variance using PSNR value. Figure 6.2 (a) is used as the training and testing image. For exponents less than 5, the computational complexity of our method is about half that of Porikli's method [77] with Taylor series approximation up to the third derivative, because at most four-pass Gaussian filtering of the exponentiation of the original image (exponent equal to 2 to 5) is required in our method, but seven passes are required in [77]. The computational complexity of the method in [56] is linearly related to the number of PBFICs used. The method in [56] can be as efficient as ours if only two PBFICs (requiring four-pass Gaussian filtering) are used. However, using only two PBFICs results in visible quantization artifacts as will be shown in Sections 6.3 and 6.4.

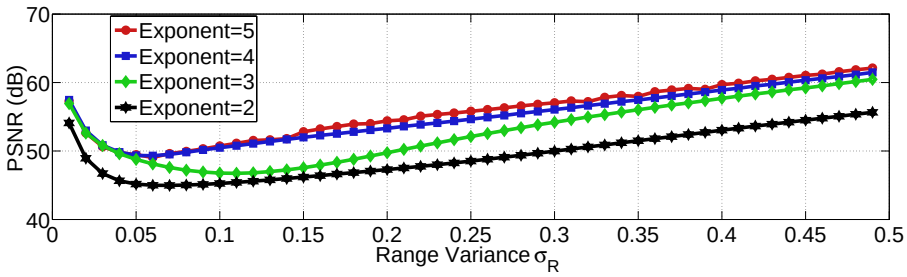
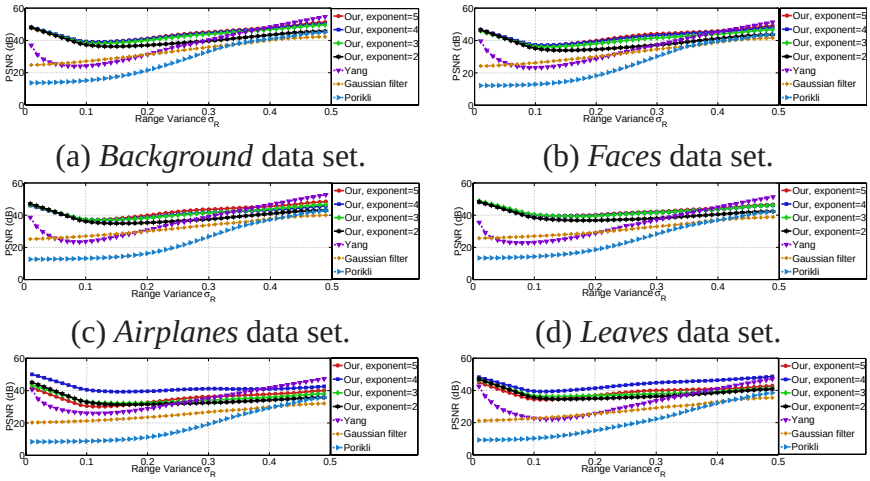


Figure 6.4: Quantitative evaluation of our method with exponents from 2 to 5 w.r.t. the range variance using PSNR value.

6.3 Experiments

In this section, I quantitatively compare our learning-based bilateral filtering method with the exact bilateral filtering using the Caltech dataset [89]. All the experiments conducted in this section and Section 6.4 use uniform range variance value for standard comparison. A model is first obtained via SVM training using one of the 450 images in the *background* category in [89], and then tested on a total of 3638 images from six categories in [89]. The mean PSNR values for different categories in [89] are presented in Figure 6.5. As discussed in Section 6.2.2,



(a) *Background* data set. (b) *Faces* data set.

(c) *Airplanes* data set. (d) *Leaves* data set.

(e) *Motorcycles* data set. (f) *Cars* data set.

Figure 6.5: PSNR accuracy w.r.t. the range variance for the Caltech data sets.

up to the 3^{rd} derivative Taylor series expansion is used for Porikli's method [77] and two PBFICs are used for Yang's method [56]. As can be seen in Figure 6.5, our method outperforms the other methods for low range variance values, where Porikli's method is invalid and Yang's method has quantization errors. For high range variance values, Gaussian filtered responses (orange curves) are very similar to bilateral filtered responses, which degrades the importance of all $O(1)$ bilateral filtering methods. Note that the accuracy of our method generally increases as the exponent increases. However, for exponents up to 5, our method has the over-fitting problem for the *motorcycles* and *cars* categories as shown in the red curves in Figure 6.5 (e) and (f). The sensitivity of our method to the choice of training image is relatively low, as proved in Figure 6.5. The same training image is used for six different categories of data sets containing 3638 different images. The blue curves in (d)-(f) show that the PSNR accuracy of our method with exponent $= 4$ is generally larger than 40 dB. It is assumed [62] the PSNR values above 40 dB often corresponds to almost invisible differences. The PSNR values in (a)-(c) are lower than those in (d)-(f) but still lager than 37 dB.

Note that it is safe to use a down-sampled version of the original image for computing the Gaussian filtered responses of its powers with almost no quality loss if both down-sampled versions are used for training and predicting. All the experiments presented in the chapter use nearest-neighbor down-sampling. The running time for the CPU and GPU implementations of all the $O(1)$ methods are

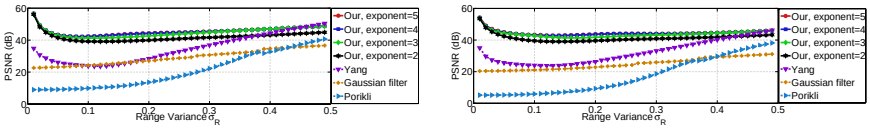
presented in Table 6.1. Note that for a 1 MB image, the speed of our fastest GPU implementation is about 473 frames per second on an Nvidia Geforce 8800GTX GPU, which is much faster than recursive Gaussian (122 fps), as can be seen in Table 6.1. For the GPU implementations of our method and the recursive Gaussian filtering, 32-bit floating point textures and global memory are used.

Table 6.1: Quantitative comparison of the speed (frame per second) of the $O(1)$ bilateral filtering methods.

	Recursive Gaussian	Porikli [77]	Yang [56]	Ours, exponent=			
				2	3	4	5
CPU	5	3	1	12	8	4	3
GPU	122	NA	120	473	308	222	166

6.4 Natural Video Conferencing

As shown in [56], edge-preserving-smoothing methods provide a way to retain the salient features in HD images while removing unwanted details and noise for modern video conference systems, e.g., Halo [83]. Our method is especially suitable for this application. As shown in Figure 6.4, if the testing image is the same as the training image, there is no visible difference between the results obtained using our method and exact bilateral filter. For this application, the training step is processed at the beginning of the conference; that is, the first frame will be used for training. The model obtained is then used to smooth the rest of the video. Because during a video conference, the content of the video streams will not change dramatically, the filtering results using our method and the exact bilateral filter are very similar, as shown in Figure 6.6. The visual evaluation is presented in Figure 6.7. The spatial variance σ_S is 0.03 and the range variance σ_R is 0.15 in this experiment. But of course, the purpose of the filtering is edge-preserving smoothing, not obtaining exact response, as in bilateral filtering.



(a) IU data set. (b) MS data set.

Figure 6.6: PSNR accuracy w.r.t. the range variance for two of *Microsoft i2i* video sequences.

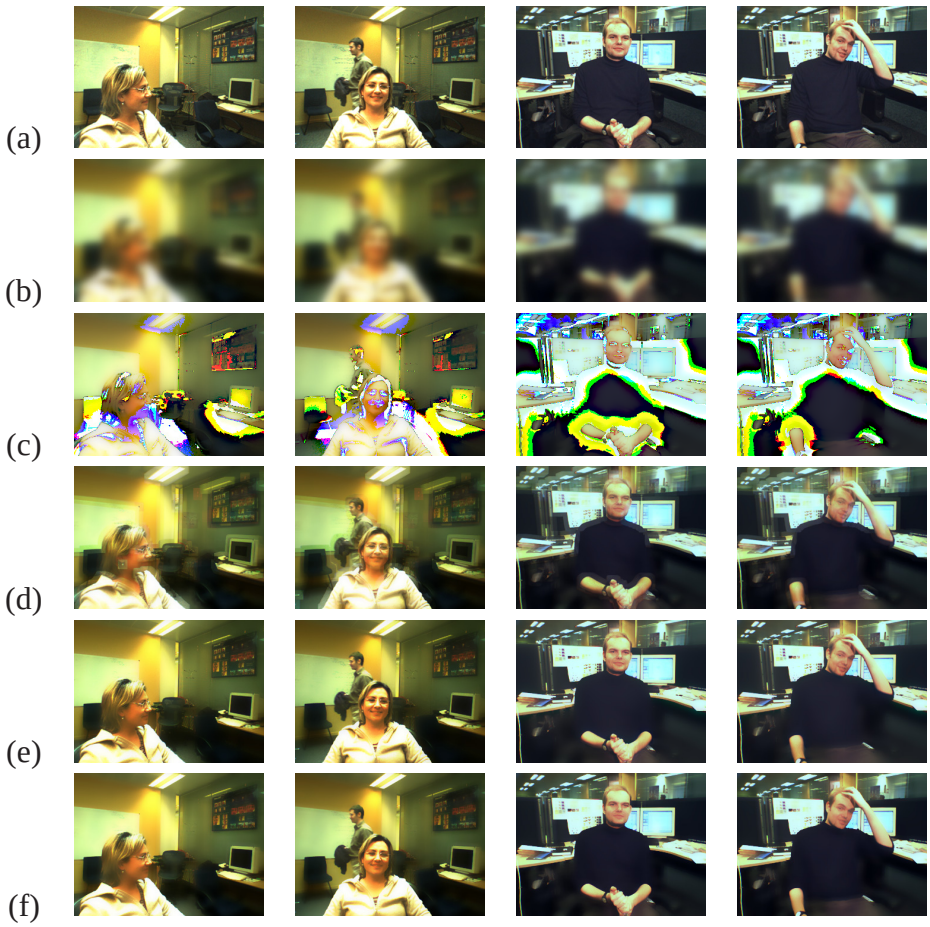


Figure 6.7: Visual comparison of the $O(1)$ bilateral filtering methods.

6.5 Discussion

This chapter describes a new $O(1)$ edge-preserving-smoothing method. To the best of our knowledge, it is the first learning-based method, and the most efficient method reported so far. Our method overcomes the common difficulties encountered in the previous $O(1)$ methods, for instance, either invalidity for low range variance Gaussian or computational time dependent on the range variance

value. Additionally, I propose a new bilateral filtering method avoiding the over-smoothing or under-smoothing artifacts in traditional bilateral filters by allowing varying range variance values crossing the image. I show that our learning-based filtering method can be directly adapted to the new bilateral filtering method since its computational complexity is independent of the range variance value.

CHAPTER 7

CONSTANT-SPACE BELIEF PROPAGATION

7.1 Introduction

Stereo vision is becoming more and more mature, especially because of publicly available performance testing such as the Middlebury benchmark [93], which allows researchers to compare their algorithms against all the state-of-the-art algorithms. The available benchmark suggests that global energy optimization methods such as belief propagation (BP) [5, 6] are essential to the state-of-the-art algorithms [12, 94, 95].

The BP algorithm works by passing messages around the graph defined by the four-connected image grid. Until recently, BP was too computationally intensive for real-time applications even for low resolution images and small number of disparity levels. Letting N be the image size, $\mathcal{L}$ be the number of disparity levels, and T be the number of iterations, the computational complexity is originally $O(4TN\mathcal{L}^2) = O(TN\mathcal{L}^2)$ [6]. Felzenszwalb and Huttenlocher [96] show that the complexity of BP can be reduced to $O(TN\mathcal{L})$ using the minimum convolution method. In addition, [96] hierarchically estimates the messages, and speeds up convergence of the original BP problem. Yang et al. [97] also propose a fast-converging BP method with computational complexity sub-linear to T by updating only the messages of the non-converging pixels. Additionally, [97] shows that the GPU implementation of the Hierarchically BP (HBP) method [96] can achieve near video rate (16 fps) with low resolution images (320×240) and small number of disparity levels (16).

Besides the slow speed, BP is also known to be memory intensive, which makes it difficult to match the data rates of the state-of-the-art cameras which can provide high-resolution stereo image pairs at video rate. For instance, the Bumblebee XB3 camera [22] can capture stereo pairs with resolution 1280×960 and more than 400 disparities at 16 FPS. A standard four-connected BP algorithm requires at

least $4N\mathcal{L}$ variables to store all the messages and $N\mathcal{L}$ for the data cost. If stored using 16-bit integers, a BP-based algorithm will required at least 4.6 GB RAM for such a stereo pair, which makes it impractical to run on ordinary computers.

Several methods have been proposed recently to reduce the memory requirement of standard BP, but mostly at the cost of increasing the running time of either message updating [98] or data cost computation [99, 100]. Additionally, most of them only reduce the memory for storing the messages [98, 99]. For standard BP, the memory used to store the data cost is about $1/4$ (or $1/3$ for HBP) of that used for messages, which is still too large for high-resolution images. One exception is the method presented in [100]. Instead of storing the data cost, the data cost is re-computed whenever it is needed.

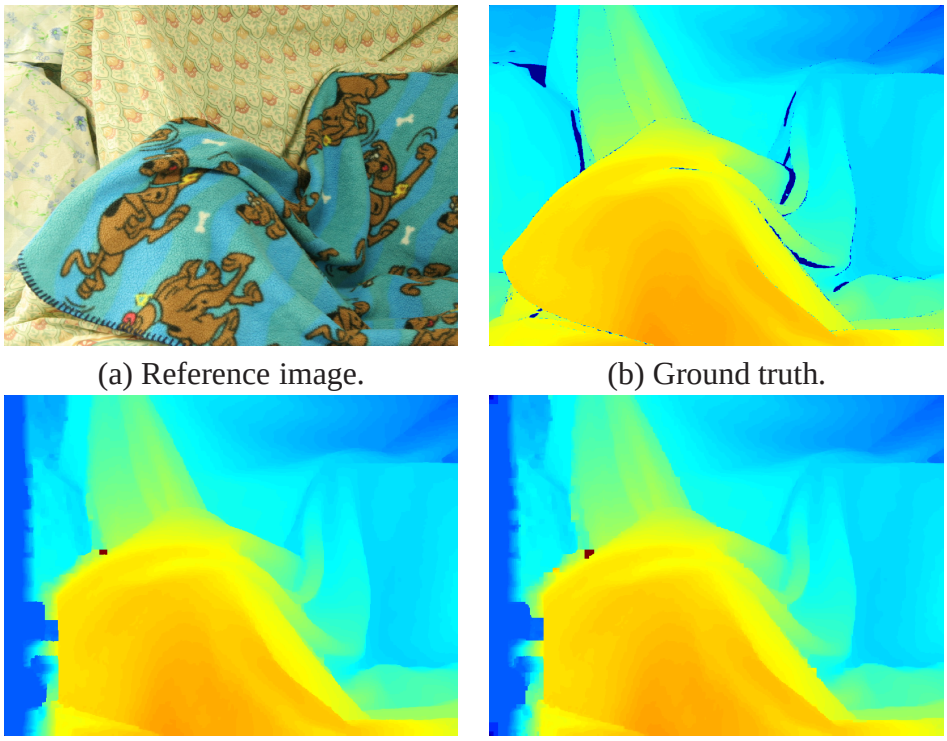
Yu et al. [98] study the feasibility of applying compression techniques to the messages in the BP algorithm to improve the memory efficiency. This method can reduce the memory cost to 12.5% for floating point precision or $30 - 50\%$ for 8-bit integer precision. However, the sequential compression and decompression operations would slow the system. Yu et al. also present an envelope point transform (EPT) based method. An extra pass of standard BP is required as a pre-processing step in this technique, which makes it slower than standard BP. This EPT method is invalid for parallel implementation because a sharing buffer is used to update the messages at every pixel. Instead of treating the messages as generic data, Wang et al. [99] preserve only a plausible subset of search space derived from the results of stereo estimates based on joint bilateral filtering [101]. The problem is that this joint bilateral filtering stereo matching method is slower than standard BP, which slows the system. Additionally, this method is invalid for parallel implementation. Liang et al. [100] split the MRF into many tiles and perform BP within each one. Global optimality can be preserved by storing the outgoing boundary messages of a tile and using them when performing BP in the neighboring tiles. This method can reduce the memory used to store the data cost. However, the data cost needs to be recomputed in each iteration and for each spatial resolution level (if hierarchical implementation is used). The methods in [98], [99] and [100] essentially sacrifice speed for memory efficiency. None of them can improve the computational complexity of standard BP. Besides, the memory complexity of these methods is still $O(\mathcal{L})$, although they do reduce the memory cost while maintaining comparable accuracy.

Here, I describe a constant space $O(1)$ BP (CSBP) method. The memory cost (including data cost) of our method is invariant to the number of disparity levels

$\mathcal{L}$. In addition, the runtime of our method is also invariant to $\mathcal{L}$ if the computation of data cost is not taken into account. To our knowledge, the presented method is the most efficient BP yet developed. This means that it will perform increasingly better as the number of disparity levels $\mathcal{L}$ increases. Given the trend toward higher-resolution images and wide-baseline stereo vision [102, 103], and consequently a larger number of disparity levels, stereo matching using BP for large $\mathcal{L}$ that is as efficient as for small $\mathcal{L}$ makes the described algorithm future-proof.

Unlike previous memory reduction methods which focus on the original spatial resolution, I hierarchically reduce the disparity range to be searched. As concluded in [98], only a small number of disparity levels and the corresponding message values at each pixel (e.g., 2 for the *Teddy* data set [93]) are needed to losslessly reconstruct the BP messages. However, there is no efficient way to compute these disparity levels. Instead of correctly locating these disparity levels, I hierarchically reduce the disparity label set as the spatial resolution increases. This method is very efficient as the number of messages at a coarser level is much smaller than that at the original level; thus, the computation is very cheap even with large number of disparity levels. If I fix the number of disparity levels at the original spatial resolution to be a constant, and repeatedly double it as the spatial resolution decreases, the memory cost of the algorithm is $O(1)$, which is linear in the image resolution and independent of $\mathcal{L}$. Our experiments conducted using the Middlebury data sets [93] show that over 99% of pixels with correct disparity values are preserved by our method on average when the number of disparity levels at the original spatial resolution is set to 2. Using this parameter setting, our memory requirement drops linearly from 3.8% of the standard HBP to 0.7% (including the storage of the data cost), as $\mathcal{L}$ increases from 50 to 300. Unlike previous methods, ours is also computationally efficient. Compared with standard HBP, the speedup factor is $6-30$ or $4.2-13.4$ if the computation of the data cost is included. Also, our algorithm lends itself to a parallel implementation. Our GPU implementation (NVIDIA Geforce 8800GTX) is about $10\times$ faster than our CPU implementation, making it the first real-time belief propagation algorithm for wide-baseline stereo matching on standard hardware that I know of. Figure 7.1 visually compares our method with standard HBP, where (c) and (d) are disparity maps obtained using standard HBP and our method, respectively. Note that there are very few noticeable differences between (c) and (d).

The limitation of our method is that it is generally less accurate than standard HBP around depth discontinuities. To solve this problem, I propose a joint bilat-



(a) Reference image. (b) Ground truth.

(c) Standard HBP. (d) Ours.

Figure 7.1: Visual evaluation on *Cloth3* data set.

eral filtering based post-processing method. The computation complexity of this method is invariant to $\mathcal{L}$. Besides, there is no additional memory requirement.

7.2 Hierarchical Belief Propagation

In this section, I briefly review the max-product BP algorithm [104] I have adopted. The max-product BP algorithm works by passing messages around the graph defined by the four-connected image grid. Each message is a vector of dimension given by the number of possible labels $\mathcal{L}$, and at each iteration, the new messages are computed as follows:

$$M^t_{\mathbf{X},\mathbf{Y}}(d) = \underset{d_{\mathbf{X}}}{\operatorname{argmin}}(E_{D,\mathbf{X}}(d_{\mathbf{X}}) + \sum_{s \in N(\mathbf{X}), \mathbf{X} \neq \mathbf{Y}} M^{t-1}_{s,\mathbf{X}}(d_{\mathbf{X}}) + h(d_{\mathbf{X}}, d)), \quad (7.1)$$

where $M^t_{\mathbf{X},\mathbf{Y}}$ is the message vector passed from pixel $\mathbf{X}$ to one of its neighbors $\mathbf{Y}$, $E_{D,\mathbf{X}}$ is the data term of pixel $\mathbf{X}$, and $h(d_{\mathbf{X}}, d)$ is the jump cost. d is the label

that minimizes the total energy for pixel $\mathbf{X}$, which contains the data term and the smoothness term

$$E_{\mathbf{X}}(d) = E_{D,\mathbf{X}}(d) + E_{S,\mathbf{X}}(d) = E_{D,\mathbf{X}}(d) + \sum_{\mathbf{Y} \in N(\mathbf{X})} M_{\mathbf{Y},\mathbf{X}}(d). \tag{7.2}$$

The common cost functions for the jump cost $h(d_{\mathbf{X}}, d)$ are based on the degree of difference between labels. In order to allow for discontinuities, the truncated linear model is commonly adopted

$$h(d_{\mathbf{X}}, d) = \rho \cdot \mathbf{min}(\mid \mathbf{d_X} - \mathbf{d} \mid, \eta), \tag{7.3}$$

where ρ is a scalar constant and η is a constant controlling when the cost stops increasing. Equation (7.3) is defined under the assumption of piecewise-smooth surfaces. Let $\mathcal{L}$ be the number of disparity levels. In this chapter, I set $\eta = \mathcal{L}/8$ and $\rho = 10$.

The global energy is observed empirically to converge after a certain number of iterations.[1] Finally, the label d that minimizes $E_{\mathbf{X}}(d)$ individually at each pixel is selected.

This standard BP algorithm is too slow to be practical. Felzenszwalb [96] proposed a hierarchical algorithm which runs much faster than the previous algorithms while maintaining comparable accuracy. The main difference between HBP and standard BP is that HBP works in a coarse-to-fine manner. The basic steps are: (a) initialize the messages at the coarsest level to all zeros, (b) apply BP at the coarsest level to iteratively refine the messages, (c) use refined messages from the coarser level to initialize the messages for the next level. Specifically, if $\mathbf{X}$ is a pixel at a coarser level, and its corresponding pixels at the finer level are $\mathbf{X}'_i$, $i \in [1, 4]$ as shown in Figure 7.2, then

$$E_{D,\mathbf{X}} = \sum_{i \in [1,4]} E_{D,\mathbf{X}'_i}, \tag{7.4}$$

$$M_{\mathbf{X}'_i,\mathbf{Y}'_{i,j}} = M_{\mathbf{X},\mathbf{Y}_j}, i, j \in [1, 4], \tag{7.5}$$

where $Y'_{i,j}$ are the four neighbors of pixel $\mathbf{X}'_i$, and $\mathbf{Y}_j$ are the corresponding four neighbors of pixel $\mathbf{X}$. For instance, assuming $Y'_{i,j}$ is the upper pixel of $\mathbf{X}'_i$, then $\mathbf{Y}_j$ is also the upper pixel of $\mathbf{X}$. Hence the number of messages at the finer lever is

[1] Loopy BP is not guaranteed to converge.

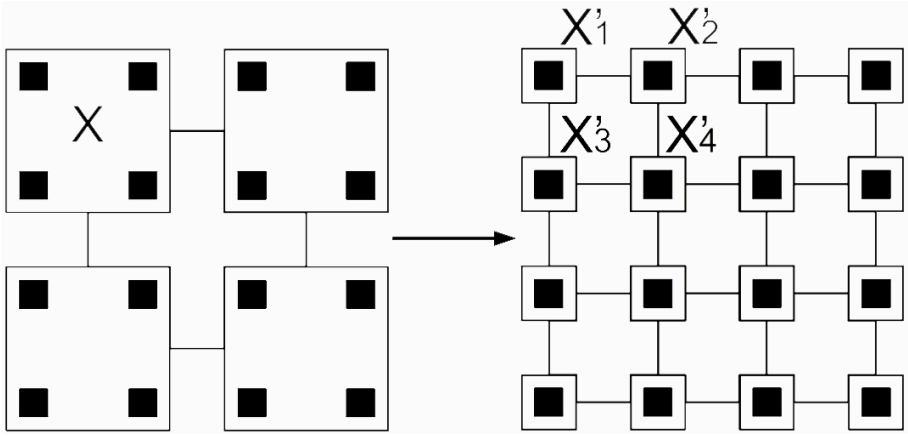


Figure 7.2: Illustration of two levels in the coarse-to-fine method.

four times that at the coarser level. As a result, the messages will be hierarchically refined while the data term stays unchanged, since the data term for the coarser level is the sum of the corresponding four data terms of the finer level as in Eqn. (7.4). Finally, the refined messages and the data term are used to compute the total energy in Eqn. (7.2).

Two main parameters $\mathcal{S}$ and T define the behavior of this HBP algorithm, $\mathcal{S}$ is the number of levels and T is the number of iterations at each level. In the chapter, I experimentally choose $\mathcal{S} = 5$ and $T = 5$.

7.3 Constant-Space Belief Propagation (CSBP)

I first give a detailed description of our constant-space algorithm in Section 7.3.1, then analyze the complexity in Section 7.3.2 and conclude that the memory required by our algorithm is linear in the image resolution but independent of the maximum disparity $\mathcal{L}$. If the computation of the data term is excluded, the running time of the algorithm is also independent of $\mathcal{L}$. I next evaluate the accuracy of our method by comparing it with standard HBP in Section 7.3.3, and discuss the limitations of our method in Section 7.3.4. I finally present an efficient post-processing method to improve the reconstruction quality in Section 7.3.5.

7.3.1 Detailed Algorithm

Our approach is inspired by the conclusion in [98] that, on average, only a small number of disparity levels and the corresponding message values are needed at each pixel (e.g., 2 for the *Teddy* [93] data set) to losslessly reconstruct the BP messages. This suggests that the message updating step involves much redundant computation. It is computationally expensive to estimate the disparity levels as an extra pass of standard BP will be required. Additionally, the number of required disparity levels will be different from pixel to pixel. As a result, the method presented in [98] is slower than standard BP, and can only reduce the memory cost of BP to about 12.5%.

In our approach, the problem is solved in another manner. I perform BP in a coarse-to-fine manner similar to [96] as presented in Section 7.2. The messages are updated at each level, and then propagated to the finer level and refined there. However, from the conclusion in [98], I know that only a few disparity levels are required. As result, I not only apply the coarse-to-fine scheme to the spatial domain, but also to the depth domain. Specifically, I gradually reduce the number of disparity levels as the messages propagate from coarsest level to original level.

Our CSBP approach is summarized in Alg. 2, which is the same as standard HBP except for steps 1, 2, 8, 9 and 10. Steps 1 and 8 show that unlike standard HBP, where the data term is computed once and stored hierarchically, our method re-computes the data term at each level (but not for each iteration). The disadvantage of our data cost computation method is that there is redundant computation and the speed is a bit slower ($9/8$, as described in Section 7.3.2) than standard HBP. However, the advantage is that the required memory for storing the data term will not depend on the maximum disparity $\mathcal{L}$. This advantage is very important for high-resolution images and wide-baseline stereo matching. For a 1280×960 image pair with 400 disparities, standard HBP requires about 1.5 GB RAM to store the data term, which is a big cost for ordinary computers and impractical for embedded systems. Step 9 in Alg. 2 is also different from standard HBP where the energy is computed only once at the fine level. Steps 2 and 10 show that unlike standard HBP where the number of disparity levels is the same for every spatial resolution, our method gradually reduces the number of disparity levels as the spatial resolution increases. Recall that BP works by looking for fixed points of the message update rule. Intuitively, the closer the messages are to the fixed points, the fewer the required disparity levels. In the extreme case, if

the messages are the same as the fixed points, then only a single disparity level is needed for each pixel, and it is the estimated disparity value. Thus the intuition of reducing the search range hierarchically is based on the behavior of HBP, which updates messages hierarchically, and refines the messages to approach the fix points.

Algorithm 2 Constant-Space Belief Propagation

1: Compute the data term E_D at the coarsest level (level $\mathcal{S}-1$): $O(N\mathcal{L})$.
2: Only $k_{\mathcal{S}-1}$ smallest data costs are selected and passed to the step 5 at each pixel. The other data costs and the corresponding disparity levels are treated as outliers: $O(N\mathcal{L}/8)$.
3: Initialize messages to zero.
4: **for** $s = \mathcal{S}-1$ to 0 **do**
5: -Iteratively update the message vector at each pixel and for each disparity candidate according to Eqn. 7.1: $O(400N)$.
6: **if** $s > 0$ **then**
7: Initialize the messages for the next level using Eqn. 7.5 and the selected disparity levels.
8: Compute the data term for the next level using the selected disparity levels: $O(3.5N)$.
9: Compute the total energy at each pixel and for each disparity candidate according to Eqn. 7.2: $O(3.5N)$.
10: Select $k_{s-1} = k_s/2$ disparity levels and message values corresponding to the k_{s-1} smallest energy values and pass them to step 5 at each pixel. The other disparity levels and messages are treated as outliers and won't be taken into account. *Note that the size of the message vector and the data term at each pixel will be reduced by half at this step*: $O(32N)$.
11: **else**
12: Compute the total energy at each pixel and each disparity candidate according to Eqn. 7.2.
13: **end if**
14: **end for**
15: The disparity value that minimizes the total energy individually at each pixel is selected.

7.3.2 Speed and Memory Analysis

If I set the number of disparity levels at the fine level to be $k_0 = 2$, which is independent of the maximum disparity $\mathcal{L}$, then the computational complexity and memory requirement of Alg. 2 are invariant to $\mathcal{L}$ except for steps 1 and 2. The

required memory in steps 1 and 2 is actually also invariant to $\mathcal{L}$ since only $k_{\mathcal{S}-1} = k_0 2^{\mathcal{S}-1} = 2^{\mathcal{S}}$ variables are required to be stored at each pixel location. Hence, our method requires only constant memory, and if the computation of the data term is excluded, our method is also constant-time. Constant-space and constant-time means that the complexity of the algorithm remains the same even if the disparity range $\mathcal{L}$ becomes very large. I next present the detailed analysis of the complexity of each step in Alg. 2. The computational complexities of each step are shown in blue text at the end of that step in Alg. 2.

The main computation of Alg. 2 is the iterative message updating at step 5. However, because the selected disparity levels may be different for neighboring pixels, the minimum convolution method presented in [96] cannot be directly used to update a message vector; thus, the computational complexity of step 5 in Alg. 2 is $O(k_s^2)$ [6]. Assuming the number of spatial resolution levels $\mathcal{S} = 5$ and the number of iterations at each scale is $T = 5$, the computational complexity of updating the messages is $O(\sum_{s=\{0,\cdots,\mathcal{S}-1\}} T \cdot 4(N/4^s)k_s^2) = O(T \cdot 16\mathcal{S}N) = O(T \cdot 80N) = O(400N)$ according to [6]. In HBP, the variables used to store the messages can be repeatedly used at different levels. Our method inherits this property, and requires $4Nk_0 = 8N$ variables. Simultaneously, our method requires extra $Nk_0 = 2N$ variables to store the selected disparity levels. So our message updating method requires a total of $10N$ variables.

Compared to standard HBP, our message updating method requires three extra steps: steps 8, 9 and 10 in Alg. 2. The computational complexity of step 8 is the same as step 9, which is $O(\sum_{s=\{1,\cdots,\mathcal{S}-2\}}(N/4^s)k_{s+1}) = O(3.5N) = O(N)$. Step 8 requires at most $Nk_0 = 2N$ variables which can be borrowed from step 2 to store the data term, and step 9 requires at most $k_{\mathcal{S}-1} = 32$ variables to store the energy values at each pixel. These 32 variables can be repeatedly used at each pixel; thus the extra memory requirement is tiny compared to the other steps. Step 10 is mainly an operation of selecting the k_s smallest values from k_{s+1} values, and can be computed directly in $O(k_s k_{s+1})$ or $O(k_s \log(k_{s+1}))$ time using heap sort. For simplicity, I assume the complexity of this operation is $O(k_s k_{s+1})$. The computational complexity of step 10 is then $O(\sum_{s=\{1,\cdots,\mathcal{S}-2\}}((0.25)^s N)k_s k_{s+1}) = O(32N) = O(N)$. There is no extra memory requirement in step 10.

To sum up, if the number of iterations is $T = 5$, the computational complexity of message updating is $O(400N)+O(3.5N)+O(3.5N)+O(32N) = O(439N) = O(N)$, and the total number of variables required is $10N$. Hence, the computational and memory complexity of our message updating method scales linearly

with the image resolution N. Although our method requires the data term to be re-computed at every level (step 8 in Alg. 2), its computational complexity is $O(3.5N)$, which is only a very small portion of the whole message updating process ($O(439N)$).

However, the computation of the data term at the coarsest level (step 1 and 2 in Alg. 2) scales linearly with $\mathcal{L}$ and may become the bottleneck of the algorithm. Specifically, the computational complexity of step 1 and 2 is $O(N\mathcal{L}) + O((0.25)^{\mathcal{S}-1}N \cdot k_{\mathcal{S}-1} \cdot \mathcal{L}) = O(N\mathcal{L}) + O(\frac{1}{8}N\mathcal{L}) = O(N\mathcal{L})$. Thus if $\mathcal{L}$ is large, the runtime of step 1 and 2 is comparable to the runtime of the messages updating steps (step $4-14$). However, the number of variables used to stored the data term is $k_{\mathcal{S}-1} \cdot (0.25)^{\mathcal{S}-1}N = N/8$. This is much smaller than $2N$, which is the largest number of variables used in step 8. Thus step 8 can share memory with steps 1 and 2 in Alg. 2, and there is no additional memory requirement for steps 1 and 2.

7.3.3 Performance Evaluation

I now quantitatively evaluate the accuracy of our method using standard test suite [93]. I numerically compare the disparity maps obtained using standard HBP and our CSBP method, and the results are summarized in Table 7.1. As can be seen, the accuracy is well preserved in our method. The percentage of pixels whose correct disparity values are preserved by our method is generally larger than 99% as shown in the last row in Table 7.1. I next evaluate the efficiency of our method with respect to the number of disparity levels $\mathcal{L}$. I use the *Cloth3* data set [93] for this experiment. The reference image is presented in Figure 7.1 (a). The resolution of the image is 800×600. The ground-truth disparity map is presented in Figure 7.1 (b). Image (c) shows the disparity map obtained using standard HBP, and (d) is the disparity map obtained using our method. $\mathcal{L}$ is set to 150 to obtain (c) and (d). Images (c) and (d) are visually very similar, which proves the accuracy of our method. Quantitative evaluation of both methods using the ground-truth disparity map presented in Figure 7.1 (b) in terms of $\mathcal{L}$ is presented in Table 7.2.

To demonstrate the efficiency of our method, I compare the runtime and memory requirement of different BP methods[2] in Figure 7.3 and 7.4. *Cloth3* data set [93] is used in this experiment. As shown in Figure 7.3 (a), our method is independent of $\mathcal{L}$ if the data cost computation is excluded while the runtimes of other

[2]The implementation of [98] is provided by T. Yu, [99] by L. Wang, and [100] from the author's website.

Table 7.1: Quantitative evaluation of the performance of the standard HBP and our method.

Algorithm	Data Set			
	Tsukuba	Venus	Teddy	Cones
Standard HBP	1.80%	1.22%	10.4%	5.61%
Ours	2.00%	1.48%	11.1%	5.98%
Preserved	99.8%	99.7%	99.2%	99.6%

Table 7.2: Quantitative evaluation on *Cloth3* data set. The numbers in the first two rows are the percentage of pixels with misestimated disparities.

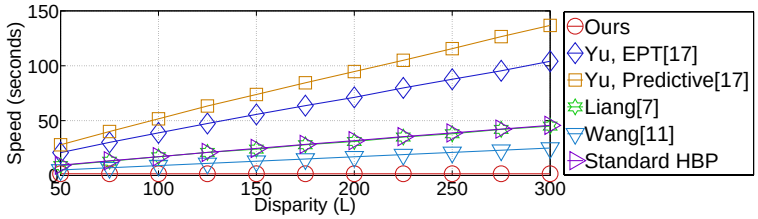
$\mathcal{L}$	100	125	150	175	200	225	250
Standard HBP	20.7%	12.3%	12.3%	12.4%	12.4%	14.1%	19.4%
Ours	21.2%	12.7%	12.8%	12.8%	12.8%	14.4%	19.4%
Preserved	99.4%	99.5%	99.5%	99.5%	99.5%	99.6%	99.9%%

methods scale linearly with $\mathcal{L}$ and are generally slower than standard HBP. Figure 7.3 (b) shows that the speedup factor of our method is $6 - 30$ as $\mathcal{L}$ increases from $50 - 300$. Figure 7.3 (c) compares the speed of the methods when the data cost computation is included. Note that the runtimes of Wang [99] and Liang's [100] methods obviously increase (by comparing to standard HBP), and are larger than standard HBP. Figure 7.4 compares the memory requirements. Apparently, our method requires constant memory and the other methods are linear in $\mathcal{L}$.

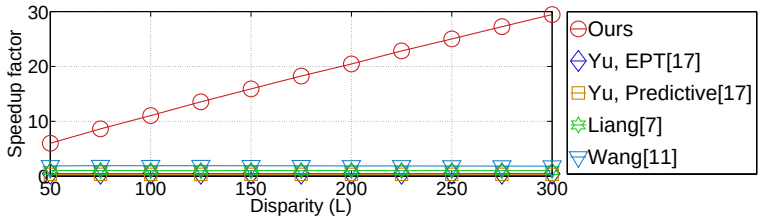
7.3.4 Limitations

One of the limitations of our CSBP method is that the runtime for updating the messages (step 5 in Alg. 2) is the same for every spatial resolution ($s = 0, \cdots, \mathcal{S}-1$). However, the runtime of standard HBP decreases quadratically with increasing s since the number of disparity levels is the same for all s.

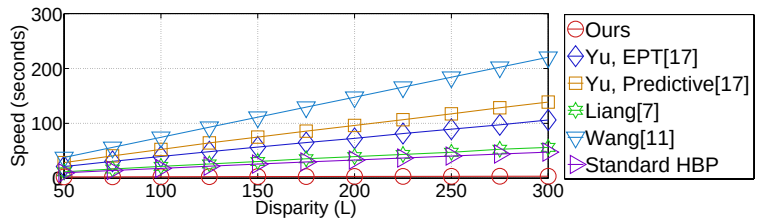
Another limitation is that the accuracy of our method is generally lower than standard HBP around depth discontinuities, since it essentially quantizes the disparity search range of standard HBP. Such an example is presented in Figure 7.7 on p. 94. Image (a) is the reference image, (b) is the ground truth, (c) is the disparity map obtained using standard HBP, and (d) is the disparity map obtained



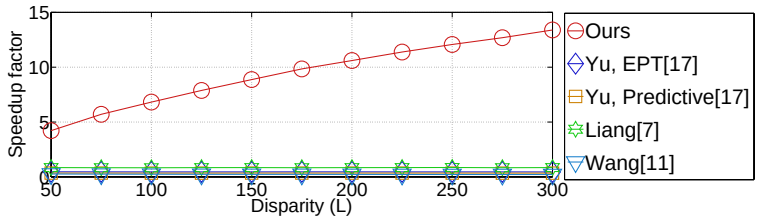
(a) Runtime.



(b) Speedup factor.



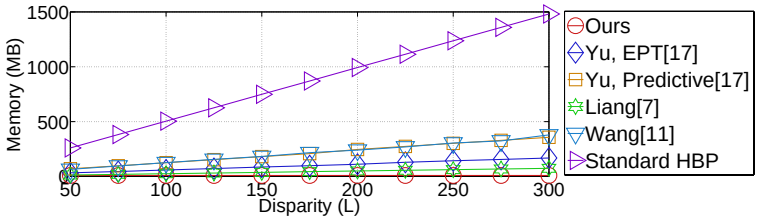
(c) Runtime (including data cost computation).



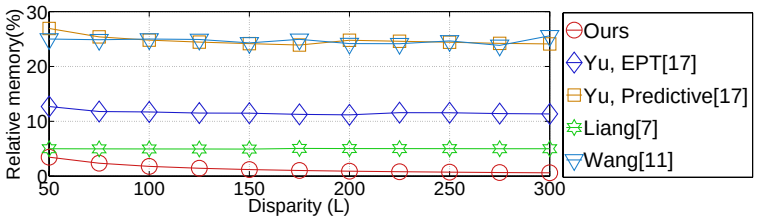
(d) Speedup factor (including data cost computation).

Figure 7.3: Comparison of the runtimes of different BP methods using *Cloth3* data set.

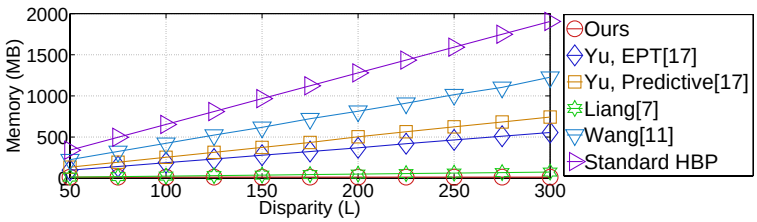
using our CSBP method in Alg. 2. The close-up of the white and red rectangles is provided in the upper left and bottom left corners. Apparently, standard HBP can better preserve the depth edges than our method, as shown in (c) and (d). For instance, the pixel marked by a red circle inside the white rectangles has incorrect disparity values in (d). This is actually because the ground-truth disparity value is not selected at the coarsest level (step 2 in Alg. 2) as shown in Figure 7.5, where the blue circle is the cost value corresponding to the ground-truth disparity value. According to Figure 7.5, it is not selected at step 2 in Alg. 2, which selects $k_{\mathcal{S}-1} = 32$ disparity values corresponding to the global minimum data cost values



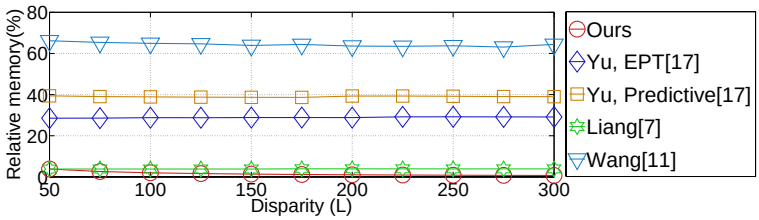
(a) Memory requirements.



(b) Relative memory.



(c) Memory requirements (including data cost).



(d) Relative memory (including data cost).

Figure 7.4: Comparison of the memory requirements of different BP methods using *Cloth3* data set.

at each pixel. Apparently, if the correct disparity value is not selected at the coarsest level (step 2), the estimated disparity value is incorrect. An obvious solution is to increase the number k_s of selected disparity levels at each spatial resolution level s, that is, reduce the quantization amount. However, the computational and memory costs will also increase. Another solution is to modify the selection algorithm at step 2 in Alg. 2 to increase the chance of including the ground-truth disparities in the selected disparity set. Instead of using global minima, I first select the local minima, and then global minima if the number of local minima is less than $k_{\mathcal{S}-1}$. The selected values are shown as cyan points in Figure 7.6. As

can be seen, the ground-truth disparity value (blue circle) is selected. The final disparity map is presented in Figure 7.7 (e) which shows that this local-minima-first selection method is better than the global minima selection method around depth edges. The local-minima-first selection method is actually a bit faster than the global method and does not require extra storage. However, this method is not suitable for repeated texture, because the number of local minima may be large and most of them are far away from the ground truth.

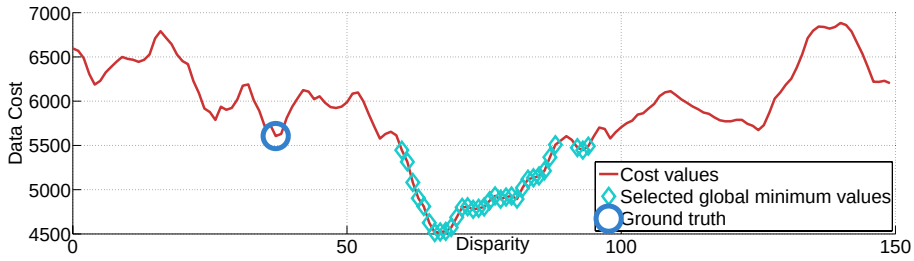


Figure 7.5: Cost profile of the point in the red circle in Figure 7.7 (a). The cyan points are the global minimum values selected at step 2 in Alg. 2, and the blue circle is the value corresponding to the ground-truth disparity, and is unselected.

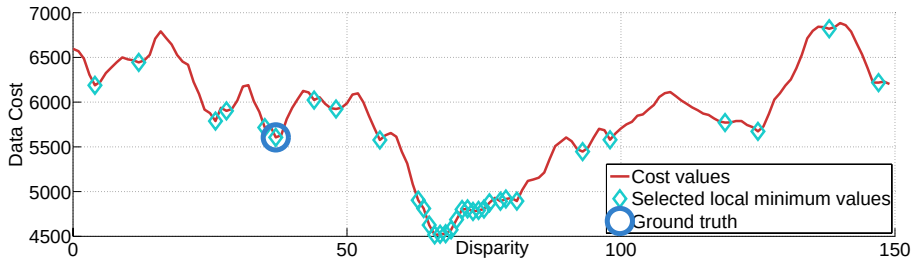
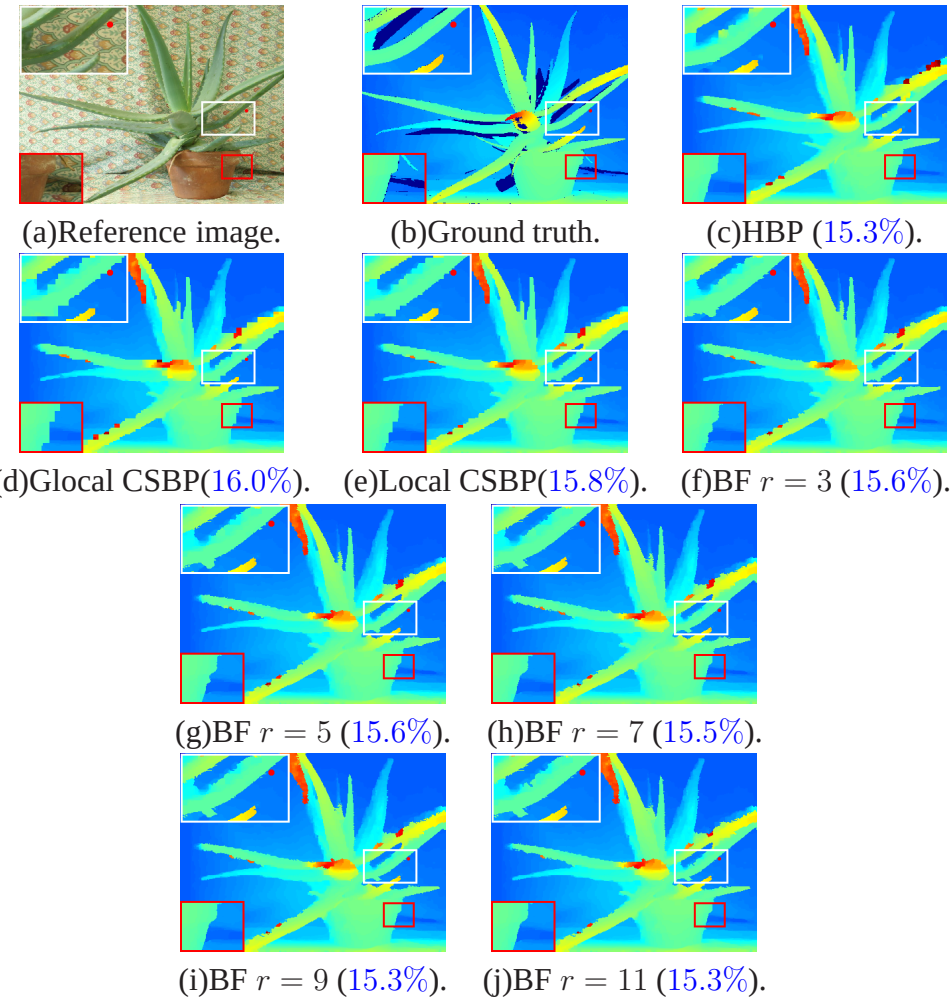


Figure 7.6: Cost profile of the point in the red circle in Figure 7.7 (a).

7.3.5 A Constant-Time Constant-Space Post-Processing Method

Visually, the accuracy of the disparity values around depth discontinuities in Figure 7.7 (e) are still less than (c), which is obtained using standard HBP. To further improve the reconstruction quality, [87] demonstrates that joint bilateral filtering can be used to filter the final energy, which helps to preserve the depth discontinuity under the assumption that color discontinuity is a strong indicator of depth discontinuity. However, the runtime of this method scales linearly with $\mathcal{L}$, and generally too slow when $\mathcal{L}$ is large. I thus modify it as a post-processing method



(a)Reference image. (b)Ground truth. (c)HBP (15.3%).
(d)Glocal CSBP(16.0%). (e)Local CSBP(15.8%). (f)BF $r = 3$ (15.6%).
(g)BF $r = 5$ (15.6%). (h)BF $r = 7$ (15.5%).
(i)BF $r = 9$ (15.3%). (j)BF $r = 11$ (15.3%).

Figure 7.7: Evaluation around depth edges.

which is invariant to $\mathcal{L}$. Let the disparity map obtained from CSBP be D, reference image be I, and bilateral filter radius be r. For a typical pixel $p = \{\mathbf{x}, \mathbf{y}\}$, assume $\vec{d}_p = \{D(\mathbf{x}-1, \mathbf{y}), D(\mathbf{x}, \mathbf{y}-1), D(\mathbf{x}+1, \mathbf{y}), D(\mathbf{x}, \mathbf{y}+1)\}$, $\vec{\mathbf{s}}_p = \{\mathbf{x}-r, \cdots, \mathbf{x}+r\}$, $\vec{\mathbf{t}}_p = \{\mathbf{y}-r, \cdots, \mathbf{y}+r\}$. I update the disparity map sequentially as follows:

$$D(\mathbf{x}, \mathbf{y}) = \underset{d \in \vec{d}_p}{\operatorname{argmin}} \frac{\sum_{\mathbf{s} \in \vec{\mathbf{s}}_p} \sum_{\mathbf{t} \in \vec{\mathbf{t}}_p} W(\mathbf{s}, \mathbf{t}) C(\mathbf{s}, \mathbf{t}, d)}{\sum_{\mathbf{s} \in \vec{\mathbf{s}}_p} \sum_{\mathbf{t} \in \vec{\mathbf{t}}_p} W(\mathbf{s}, \mathbf{t})}, \tag{7.6}$$

where

$$W(\mathbf{s}, \mathbf{t}) = exp(-\frac{||I(\mathbf{x}, \mathbf{y}), I(\mathbf{s}, \mathbf{t})||_2}{2\sigma_R^2}) \cdot exp(-\frac{(\mathbf{x}-\mathbf{s})^2 + (\mathbf{y}-\mathbf{t})^2}{2r^2}) \tag{7.7}$$

$$C(\mathbf{s}, \mathbf{t}, d) = \min(\lambda\mathcal{L}, \mid D(\mathbf{s}, \mathbf{t}) - d \mid), \tag{7.8}$$

and $\lambda = 0.2$ is a constant to reject outliers. Setting $\sigma_R = 10$, the post-processed disparity maps of Figure 7.7 (e) with different filter radius $r = 3 - 11$ are presented in Figure 7.7 (f)-(j). Note that this post-processing method is invariant to $\mathcal{L}$ and only valid for pixels around depth discontinuities. It is thus required to check whether a pixel is on a depth edge before processing it. As a result, only a small fraction of the pixels will be processed, and the runtime will depend not only on the image resolution but also on the image structure. However, it scales linearly with the filter size ($(2r+1)^2$) excluding the time used to locate depth edges. The exact runtimes (seconds) to obtain the disparity maps shown in Figure 7.7 (f)-(j) are 0.05, 0.12, 0.21, 0.32 and 0.47, respectively. Apparently, this post processing method is very efficient even for high resolution disparity maps. The blue numbers below Figure 7.7 (c)-(j) are the percentages of pixels with misestimated disparities, which numerically prove that the proposed post processing method can effectively improve the reconstruction quality.

7.4 Discussion

In this chapter, I propose a constant-space BP method. The memory cost (including data cost) of our method is invariant to the number of disparity levels. Besides, the runtime of our method is independent of the number of disparity levels if the computation of data cost is excluded. Experiments using Middlebury data sets [93] (Tables 7.1, 7.2 and Figure 7.7) show that our method results in over 99% pixels, on average, having correct disparity values. The limitation of our method is that it is less accurate than standard HBP around depth discontinuities. To solve this problem, I propose a very efficient post-processing method. The computational complexity of this method is independent of the number of disparity levels, and there is no additional memory requirement.

CHAPTER 8

CONCLUSION AND FUTURE WORK

This dissertation focuses on stereo vision based 3D reconstruction. Unlike available vision based 3D capturing systems, this system is much more efficient and more robust for matching low-textured regions and specular highlights.

This dissertation addresses the problem of how to robustly reconstruct the 3D shape in the lack of texture and the presence of strong specularities. For matching low-textured regions, I develop a new image transform - epipolar distance transform, which captures the local image structure by computing the ratios of lengths along the epipolar lines. The ratio of the distances is invariant to affine transformation, and thus can be used as a matching invariant for stereo vision. Unlike image intensity/color, this invariant is robust for matching low-texture regions. This dissertation also carefully analyzes the interaction between illumination, reflectance and shape to avoid the incorrect matches due to specular highlights. This dissertation uses a new correspondence matching invariant, Illumination Chromaticity Constancy, to estimate illumination chromaticity. Using as few as a single image, the estimated illumination chromaticity is then used together with a novel bilateral filtering based highlight removal method to separate the diffuse and specular components which are then fed to an iterative optimization module for simultaneously estimating the depth and diffuse reflection via multi-view stereo matching and depth map fusion.

This dissertation also addresses the problem of algorithm complexity. Two fast local denoising algorithms and a fast global energy minimization algorithm are proposed. These algorithms are very useful for specular highlight removal and stereo matching.

Many questions remain to be answered. How should saturated highlight regions be handled? What if the chromatic surfaces assumption does not hold (e.g., human faces)? How to model objects under water? I hope this dissertation is a step forward to eventually solve these problems. I envision that in the near future, the virtual world will be as real as the current world in front of our eyes.

REFERENCES

[1] D. Scharstein and R. Szeliski, "A taxonomy and evaluation of dense two-frame stereo correspondence algorithms," *International Journal of Computer Vision*, vol. 47, no. 1/2/3, pp. 7–42, 2002.

[2] M. Brown, D. Burschka, and G. D. Hager, "Advances in computational stereo," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 8, pp. 993–1008, 2003.

[3] K.-J. Yoon and I.-S. Kweon, "Adaptive support-weight approach for correspondence search", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 4, pp. 650–656, 2006.

[4] C. Zitnick, S. Kang, M. Uyttendaele, S. Winder, and R. Szeliski, "High-quality video view interpolation using a layered representation," in *ACM Siggraph*, 2004, pp. 600–608.

[5] W. T. Freeman, E. Pasztor, and O. T. Carmichael, "Learning low-level vision," *International Journal of Computer Vision*, vol. 40, no. 1, pp. 25–47, 2000.

[6] J. Sun, N. Zheng, and H. Y. Shum, "Stereo matching using belief propagation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 7, pp. 787–800, 2003.

[7] Y. Boykov, O. Veksler, and R. Zabih, "Fast approximate energy minimization via graph cuts," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, no. 11, pp. 1222–1239, 2001.

[8] D. Terzopoulos, "Regularization of inverse visual problems involving discontinuities," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 8, no. 4, pp. 413–242, 1986.

[9] H. Tao, S. Harpreet, and R. Kumar, "Segment-based stereo matching using belief propagation and a self-adapting dissimilarity measure," in *IEEE International Conference on Computer Vision*, pp. 532–539, 2001.

[10] M. Bleyer and M. Gelautz, "A layered stereo algorithm using image segmentation and global visibility constraints," in *International Conference on Image Processing*, pp. 2997–3000, 2004.

[11] A. Klaus, M. Sormann, and K. Karner, "Segment-based stereo matching using belief propagation and a self-adapting dissimilarity measure," in *International Conference on Pattern Recognition*, pp. 15–18, 2006.

[12] Q. Yang, L. Wang, R. Yang, H. Stewenius, and D. Nister, "Stereo matching with color-weighted correlation, hierachical belief propagation and occlusion handling," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 3, pp. 492–504, 2009.

[13] M. Coughlan and L. Yuille, "Manhattan world: Compass direction from a single image by Bayesian inference," in *IEEE International Conference on Computer Vision*, pp. 941–948, 1999.

[14] Y. Furukawa, B. Curless, S. Seitz, and R. Szeliski, "Manhattan-world stereo," in *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1422–1429, 2009.

[15] S. Coorg and S. Teller, "Extracting textured vertical facades from controlled close-range imagery," in *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 625–632, 1999.

[16] T. Werner and A. Zisserman, "New techniques for automated architectural reconstruction from photographs," in *European Conference on Computer Vision*, pp. 541–555, 2002.

[17] F. Porikli, "Integral histogram: a fast way to extract histograms in Cartesian spaces," in *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 829–836, 2005.

[18] N. Ahuja, "A transform for multiscale image segmentation by integrated edge and region detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 18, pp. 1211–1235, 1996.

[19] S. Todorovic and N. Ahuja, "Extracting subimages of an unknown category from a set of images," in *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 927–934, 2006.

[20] D. Comaniciu and P. Meer, "A robust approach toward feature space analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, pp. 603–619, 2002.

[21] S. Paris and F. Durand, "A fast approximation of the bilateral filter using a signal processing approach," *International Journal of Computer Vision*, vol. 81, no. 1, pp. 24–52, 2009.

[22] Point Grey Research, Inc., product overview for Bumblebee XB3. [Online]. Available: http://www.ptgrey.com/products/bbxb3/index.asp

[23] Q. Yang, R. Yang, J. Davis, and D. Nistér, "Spatial-depth super resolution for range images," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2007.

[24] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, pp. 91–110, 2004.

[25] K. S. Arun, T. S. Huang, and S. D. Blostein, "Least-squares fitting of two 3-d point sets," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 698–700, 1987.

[26] H.-C. Lee, "Method for computing the scene-illuminant chromaticity from specular highlights," *COLOR*, pp. 303–308, 1992.

[27] G. Finlayson and G. Schaefer, "Solving for colour constancy using a constrained dichromatic reflection model," *International Journal of Computer Vision*, vol. 42, no. 3, pp. 127–144, 2001.

[28] R. T. Tan, K. Nishino, and K. Ikeuchi, "Illumination chromaticity estimation using inverse-intensity chromaticity space," in *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 673–680, 2003.

[29] S. Lin and H.-Y. Shum, "Separation of diffuse and specular reflection in color images," in *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 341–346, 2001.

[30] R. Tan and K. Ikeuchi, "Reflection components decomposition of textured surfaces using linear basis functions," in *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 125–131, 2005.

[31] R. Tan and K. Ikeuchi, "Separating reflection components of textured surfaces using a single image," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 2, pp. 178–193, 2005.

[32] P. Tan, L. Quan, and S. Lin, "Separation of highlight reflections on textured surfaces," in *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1855–1860, 2006.

[33] S. P. Mallick, T. E. Zickler, D. J. Kriegman, and P. N. Belhumeur, "Beyond lambert: Reconstructing specular surfaces using color," in *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 619–626, 2005.

[34] S. P. Mallick, T. Zickler, P. N. Belhumeur, and D. J. Kriegman, "Specularity removal in images and videos: A PDE approach," in *European Conference on Computer Vision*, pp. 550–563, 2006.

[35] T. Zickler, S. P. Mallick, D. J. Kriegman, and P. N. Belhumeur, "Color subspaces as photometric invariants," in *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2000–2010, 2006.

[36] K.-J. Yoon and I.-S. Kweon, "Correspondence search in the presence of specular highlights using specular-free two-band images," in *Asian Conference on Computer Vision*, pp. 761–770, 2006.

[37] A. Blake, "Specular stereo," in *International Joint Conferences on Artificial Intelligence*, pp. 973–976, 1985.

[38] G. Brelstaff and A. Blake, "Detecting specular reflections using Lambertian constraints," in *IEEE International Conference on Computer Vision*, pp. 297–302, 1988.

[39] A. Chakrabarti, K. Hirakawa, and T. Zickler, "Color constancy beyond bags of pixels," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2008.

[40] S. Lin, Y. Li, S. Kang, X. Tong, and H.-Y. Shum, "Diffuse-specular separation and depth recovery from image sequences," in *European Conference on Computer Vision*, pp. 210–224, 2002.

[41] D. Bhat and S. Nayar, "Stereo in the presence of specular reflection," in *IEEE International Conference on Computer Vision*, pp. 1086–1092, 1995.

[42] S. Shafer, "Using color to separate reflection components," *Color Research and Application*, vol. 10, no. 4, pp. 210–218, 1985.

[43] P. Merrell, A. Akbarzadeh, L. Wang, P. Mordohai, J.-M. Frahm, R. Yang, D. Nistr, and M. Pollefeys, "Real-time visibility-based fusion of depth maps," in *IEEE International Conference on Computer Vision*, pp. 1–8, 2007.

[44] D. Scharstein and R. Szeliski, "A taxonomy and evaluation of dense two-frame stereo correspondence algorithms," *International Journal of Computer Vision*, vol. 47, no. 1-3, pp. 7–42, 2002.

[45] G. D. Finlayson, S. D. Hordley, C. Lu, and M. S. Drew, "On the removal of shadows from images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 1, pp. 59–68, 2006.

[46] R. Zhang, P.-S. Tsai, J. E. Cryer, and M. Shah, "Shape from shading: A survey," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 21, pp. 690–706, 1999.

[47] S. M. Seitz, B. Curless, J. Diebel, D. Scharstein, and R. Szeliski, "A comparison and evaluation of multi-view stereo reconstruction algorithms," in *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 519–528, 2006.

[48] R. J. Woodham, "Photometric method for determining surface orientation from multiple images," *Optical Engineering*, vol. 19, pp. 139–144, 1980.

[49] H. Jin, S. Soatto, and A. J. Yezzi, "Multi-view stereo beyond Lambert," in *IEEE Conference on Computer Vision and Pattern Recognition*, vol. 1, pp. 171–178, 2003.

[50] T. Yu, N. Xu, and N. Ahuja, "Shape and view independent reflectance map from multiple views," *International Journal of Computer Vision*, vol. 73, no. 2, pp. 123–138, 2007.

[51] T. Yu, N. Xu, and N. Ahuja, "Recovering shape and reflectance model of non-Lambertian objects from multiple views," in *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 226–233, 2004.

[52] B. T. Phong, "Illumination for computer generated pictures," *Communications of the ACM*, vol. 18, pp. 311–317, 1975.

[53] S. M. Seitz and C. R. Dyer, "Photorealistic scene reconstruction by voxel coloring," *International Journal of Computer Vision*, vol. 35, pp. 151-173, 1999.

[54] R. Koch, M. Pollefeys, and L. Gool, "Multi viewpoint stereo from uncalibrated video sequences," in *European Conference on Computer Vision*, pp. 55–71, 1998.

[55] F. Porikli, "Constant time O(1) bilateral filtering," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2008.

[56] Q. Yang, K.-H. Tan, and N. Ahuja, "Real-time O(1) bilateral filtering," in *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 557–564, 2009.

[57] R. Tan, Highlight Removal From A Single Image, 2010. [Online]. Available: http://people.cs.uu.nl/robby/

[58] M. Pollefeys, D. Nistér, J. M. Frahm, A. Akbarzadeh, P. Mordohai, B. Clipp, C. Engels, D. Gallup, S. J. Kim, P. Merrell, C. Salmi, S. Sinha, S. Sinha, B. Talton, L. Wang, Q. Yang, H. Stewénius, R. Yang, G. Welch, and H. Towles, "Detailed real-time urban 3d reconstruction from video," *International Journal of Computer Vision*, vol. 78, no. 2-3, pp. 143–167, 2008.

[59] D. Nister, "Automatic dense reconstruction from uncalibrated video sequences," *PhD dissertation, Royal Institute of Technology KTH, Stockholm, Sweden*, 2001.

[60] C. Tomasi and R. Manduchi, "Bilateral filtering for gray and color images," in *IEEE International Conference on Computer Vision*, pp. 839–846, 1998.

[61] F. Durand and J. Dorsey, “Fast bilateral filtering for the display of high-dynamic-range images,” in *ACM Siggraph*, vol. 21, no. 3, pp. 257-266, 2002.

[62] S. Paris and F. Durand, “A fast approximation of the bilateral filter using a signal processing approach,” *International Journal of Computer Vision*, vol. 81, pp. 24-52, 2009.

[63] A. Buades, B. Coll, and J.-M. Morel, “A review of image denoising algorithms, with a new one,” *Multiscale Modeling and Simulation*, vol. 4, no. 2, pp. 490–530, 2005.

[64] W. C. K. Wong, A. C. S. Chung, and S. C. H. Yu, “Trilateral filtering for biomedical images,” in *International Symposium on Biomedical Imaging*, pp. 820–823, 2004.

[65] E. P. Bennett and L. McMillan, “Video enhancement using per-pixel virtual exposures,” in *ACM Siggraph*, vol. 24, no. 3, 2005, pp. 845–852.

[66] E. P. Bennett, J. L. Mason, and L. McMillan, “Multispectral bilateral video fusion,” *IEEE Transactions on Image Processing*, vol. 16, no. 5, pp. 1185–1194, 2007.

[67] B. M. Oh, M. Chen, J. Dorsey, and F. Durand, “Image-based modeling and photo editing,” in *ACM Siggraph*, pp. 433–442, 2001.

[68] G. Petschnigg, M. Agrawala, H. Hoppe, R. Szeliski, M. Cohen, and K. Toyama, “Digital photography with flash and no-flash image pairs,” in *ACM Siggraph*, pp. 664-672, 2004.

[69] R. Ramanath and W. E. Snyder, “Adaptive demosaicking,” *Journal of Electronic Imaging*, vol. 12, no. 4, pp. 633–642, 2003.

[70] H. Winnemoller, S. C. Olsen, and B. Gooch, “Real-time video abstraction,” in *ACM Siggraph*, pp. 1221–1226, 2006.

[71] J. Xiao, H. Cheng, H. Sawhney, C. Rao, and M. Isnardi, “Bilateral filtering-based optical flow estimation with occlusion detection,” in *European Conference on Computer Vision*, pp. 211–224, 2006.

[72] P. Sand and S. Teller, “Particle video: Long-range motion estimation using point trajectories,” *International Journal of Computer Vision*, vol. 81, pp. 72–91, 2008.

[73] M. Elad, “On the bilateral filter and ways to improve it,” *IEEE Transactions on Image Processing*, vol. 11, no. 10, pp. 1141–1151, 2002.

[74] T. Q. Pham and L. J. van Vliet, “Separable bilateral filtering for fast video preprocessing,” in *International Conference on Multimedia and Expo*, pp. 454–457, 2005.

[75] B. Weiss, “Fast median and bilateral filtering,” in *ACM Siggraph*, pp. 519–526, 2006.

[76] J. Chen, S. Paris, and F. Durand, “Real-time edge-aware image processing with the bilateral grid,” in *ACM Siggraph*, pp. 103–113, 2007.

[77] F. Porikli, “Constant time O(1) bilateral filtering,” in *IEEE Conference on Computer Vision and Pattern Recognition*, 2008.

[78] P. Viola and M. Jones, “Robust real-time face detection,” in *IEEE International Conference on Computer Vision*, pp. 747–750, 2001.

[79] R. Deriche, “Recursively implementing the Gaussian and its derivatives,” in *International Conference on Image Processing*, pp. 263–267, 1992.

[80] Q. Yang, Constant space belief propagation, 2010. [Online]. Available: http://vision.ai.uiuc.edu/˜qyang6/

[81] E. Eisemann and F. Durand, “Flash photography enhancement via intrinsic relighting,” in *ACM Siggraph*, pp. 673–678, 2004.

[82] F. Crow, “Summed-area tables for texture mapping,” in *ACM Siggraph*, pp. 207–212, 1984.

[83] Hewlett-Packard, Hp halo telepresence and video conferencing solutions. [Online]. Available: http://www.hp.com/halo

[84] A. Zucker, S. Lev, and A. Rosenfeld, “Iterative enhancement of noisy images,” *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 7, pp. 435–441, 1977.

[85] L. S. Davis and A. Rosenfeld, “Noise cleaning by iterated local averaging,” *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 8, pp. 703–710, 1978.

[86] M. Tabb and N. Ahuja, “Unsupervised multiscale image segmentation by integrated edge and region detection,” *IEEE Transactions on Image Processingg*, vol. 6, pp. 642–655, 1997.

[87] Q. Yang, C. Engels, and A. Akbarzadeh, “Near real-time stereo for weakly-textured scenes,” in *British Machine Vision Conference*, pp. 80–87, 2008.

[88] Q. Yang, L. Wang, and N. Ahuja, "A constant-space belief propagation algorithm for stereo matching," in *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1458-1465, 2010.

[89] C. V. Lab, Caltech dataset. [Online]. Available: http://www.vision.caltech.edu/html-files/archive.html

[90] M. R. Cambridge, Microsoft i2i dataset. [Online]. Available: http://research.microsoft.com/en-us/projects/i2i/data.aspx

[91] V. N. Vapnik, "Statistical Learning Theory," *Wiley-Interscience*, September 1998.

[92] C.-C. Chang and C.-J. Lin, LIBSVM: a library for support vector machines, 2001. [Online]. Available: http://www.csie.ntu.edu.tw/˜cjlin/libsvm

[93] D. Scharstein and R. Szeliski, Middlebury benchmark. [Online]. Available: http://vision.middlebury.edu/stereo/

[94] A. Klaus, M. Sormann, and K. Karner, "Segment-based stereo matching using belief propagation and a self-adapting dissimilarity measure," in *International Conference on Pattern Recognition*, pp. 15–18, 2006.

[95] M. Bleyer, M. Gelautz, C. Rother, and C. Rhemann, "A stereo approach that handles the matting problem via image warping," in *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 501–508, 2009.

[96] P. Felzenszwalb and D. Huttenlocher, "Efficient belief propagation for early vision," *International Journal of Computer Vision*, vol. 70, no. 1, pp. 41–54, 2006.

[97] Q. Yang, L. Wang, R. Yang, S. Wang, M. Liao, and D. Nistér, "Real-time global stereo matching using hierarchical belief propagation," in *British Machine Vision Conference*, pp. 989–998, 2006.

[98] T. Yu, R.-S. Lin, B. S., and B. Tang, "Efficient message representations for belief propagation," in *IEEE International Conference on Computer Vision*, pp. 1–8, 2007.

[99] L. Wang, H. Jin, and R. Yang, "Search space reduction for MRF stereo," in *European Conference on Computer Vision*, pp. 576–588, 2008.

[100] C. Liang, C. Cheng, Y. Lai, L. Chen, and H. Chen, "Hardware-efficient belief propagation," in *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 80–87, 2009.

[101] L. Wang, M. Liao, M. Gong, R. Yang, and D. Nister, "High-quality real-time stereo using adaptive cost aggregation and dynamic programming," in *International Symposium on 3D Data Processing Visualization and Transmission*, pp. 798–805, 2006.

[102] E. Tola, V. Lepetit, and P. Fua, "Daisy: An efficient dense descriptor applied to wide baseline stereo," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, pp. 815–830, 2009.

[103] D. Gallup, J.-M. Frahm, P. Merrell, and M. Pollefeys, "Variable baseline/resolution stereo," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2008.

[104] Y. Weiss and W. T. Freeman, "On the optimality of solutions of the max-product belief-propagation algorithm in arbitrary graphs," *IEEE Transactions on Information Theory*, vol. 47, no. 2, pp. 736–744, 2001.

AUTHOR'S BIOGRAPHY

Yang, Qing-Xiong received the BE degree from University of Science and Technology of China (USTC) in 2004, MSc degree in Computer Science from University of Kentucky in 2007 and PhD degree in Electrical and Computer Engineering from University of Illinois at Urbana-Champaign in 2010. His research interests reside in computer vision and graphics. He won the best student paper award at MMSP 2010.